PLETL SZILVESZTER:*

Nemlineáris, dinamikus rendszerek hatékony optimalizálásának egy lehetősége

Abstract

Evolution as a process has remarkable problem-solving abilities. Today the available technical items enable the successful use of genetic algorithms as a sort of model of evolution is various fields of science. The procedure presented in this work combines the advantages of well-known soft techniques. Let it be assumed that the criterion function to be minimized has a global minimum. Let $f: D \to \Re$, $D \subseteq \Re^n$ be a criterion function. The aim is to determine the $\vartheta^* = \arg\min_{\vartheta \in D} f(\vartheta)$ global minimum. The use of the genetic algorithm in the stochastic-based search will most likely result in finding this minimum. In the further part of this work ϑ will dente the free parameters of the system. The algorithms presented in this paper can be made parallel to a great extent and thus can be largely implemented in grid systems.

Bevezető

Az evolúciónak mint folyamatnak a természetben jelentős problémamegoldó képessége van. A napjainkban létező technikai eszközök lehetővé teszik a genetikus algoritmusok, mint az evolúció egyféle modelljének sikeres felhasználását a tudomány több területén. A munkában bemutatott eljárás ötvözi az ismert szoft technikák előnyeit. Tételezzük fel, hogy a minimizálandó kritériumfüggvénynek létezik egy globális minimuma. Legyen $f: D \to \Re$, $D \subseteq \Re^n$ egy kritériumfüggvény. Célunk a $\vartheta^* = \arg\min_{\vartheta \in D} f(\vartheta)$ globális minimum meghatározása. A sztochasztikus keresésen alapuló genetikus algoritmus felhasználásával nagy valószínűséggel megtalálhatjuk ezt a minimumot. A továbbiakban ϑ a rendszer szabad paramétereit jelöli. A munkában bemutatásra kerülő algoritmus nagymértékben párhuzamosítható és így nagymértékben alkalmas grid rendszereken való implementálásra.

1. Genetikus algoritmus

Genetikus algoritmusok (GA) olyan optimalizációs eljárások, melyek a természetben fellelhető folyamatok modellezésén alapulnak. Az optimumkeresés eltér a hagyományos eljárásoktól a következőkben:

- a keresés folyamatában nem egy, hanem több paraméterhalmaz vesz részt
- a keresés több pontból indul
- a kereséshez költségfüggvényt használ, nem pedig gradienst.

A már klasszikusnak számító genetikus algoritmusok mindegyike, legyen az egy populációs vagy több populációs, magában foglalja az alábbi lépéseket:

1. Az optimalizálandó rendszer paramétereinek megfelelő kódolása.
2. Az első generáció egyedeinek meghatározása.
3. A költségfüggvény függvény kiértékelése. Alkalmassági érték (fitness) hozzárendelése az egyes egyedekhez a költségfüggvény alapján (minősítés, életképesség).
4. Az egyedek szelekciója.

* PhD – SZTE TTIK Informatikai Tanszékcsoport.

5. Az egyedek keresztezése.
6. Az egyedek mutációja.
7. Migráció (csak multipopulációs algoritmus esetén).
8. Az algoritmus 3. pontjától való ismétlés, míg el nem jutunk a keresett optimális megoldáshoz.

Az irodalomban számos variáció ismert a fenti algoritmus egyes pontjainak és hatékonyságát befolyásoló paramétereinek (populáció méret, szelekciós ráta, mutációs valószínűség stb.) definiálására. Ebben a munkában bemutatásra kerül egy olyan hibrid genetikus algoritmus, amely nemlineáris dinamikus rendszerek optimalizációjánál gyors konvergenciát eredményez.

Dinamikus rendszerek optimalizálásánál minden egyed esetében tranzienst kell meghatározni. Számítógéppel történő tranziens számítás esetén gyenge minősítési jellemzők észlelésekor az egyedhez történő rossz alkalmassági érték hozzárendelése mellett leállítjuk a számítást. Valós rendszer alkalmazása esetén biztosítani kell, hogy a gyenge minősítésű egyedek ne tudják károsítani a fizikai rendszert, vagy a technológiai folyamatot. A lágy számítási módszereknek dinamikus rendszerek irányításában való gyakorlati alkalmazásáról többek között a [2], [3,] és a [4] foglalkozik.

2. Szerzett tapasztalatokat is örökítő genetikus algoritmus

A természetben fellelhető evolúciós folyamatoknál a szülők élete alatt tanulással szerzett tapasztalatok nem öröklődnek. Az utódok paramétereit azonban befolyásolják az idősebb generációkkal való együttélés során tőlük szerzett tapasztalatok. Ez a folyamat mesterségesen a hagyományos genetikai algoritmust módosítva modellezhető [4]. Az új algoritmus, mely alkalmas nemlineáris dinamikus rendszerek optimalizálására, a következő lépésekből áll:

1. Az optimalizálandó rendszer paramétereinek megfelelő kódolása.
2. Az első generáció egyedeinek meghatározása úgy, hogy az egyes egyedek által kódolt paraméterekkel definiált rendszer elégítse ki az alapvető elvárásokat (paraméterek alsó és felső korlátja, stabilitás stb.).
3. Jósági tényező hozzárendelése (pl. $1/(1+f(\cdot))$, ahol $f(\cdot)\geq 0$) a „fiatal" egyedekhez.
4. A fiatal egyedek tanítása, az idős egyedek létrehozása.
5. A megváltozott objektív értékek kiértékelése.
6. Az egyedek szelekciója.
7. Az egyedek keresztezése.
8. Az egyedek mutációja.
9. Migráció (csak multipopulációs algoritmus esetén).
10. Az algoritmus 3. pontjától való ismétlése, míg el nem jutunk a keresett optimális megoldáshoz.

***1. ábra.* LEGA algoritmus**

Nevezzük az *1. ábrán* szereplő algoritmust szerzett tapasztalatokat is örökítő genetikus algoritmusnak (Learning-Expanded Genetic Algorithm, LEGA).

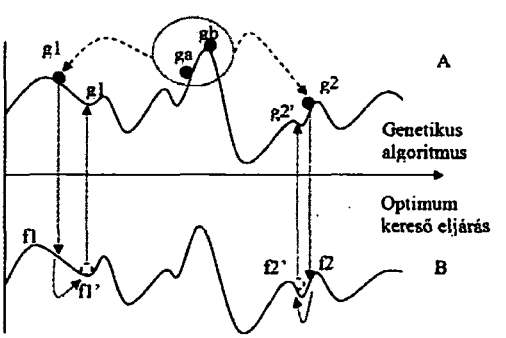
A LEGA első pontjaként említett paraméterkódolás fontos eleme az optimalizációs eljárásnak. Nevezzük a rendszer fizikai paramétereit fenotípusnak, az egyedekkel kódolt paramétereket pedig genotípusnak. Az optimalizálandó rendszer szabad paraméterértékeinek alsó és felső határát, valamint az azon belüli felbontást jól meg kell becsülni. Ehhez szakértelemre és a rendszerről szerzett előzetes információkra van szükség. Esetleg a határok és a felbontás további szabadságfokot jelenthetnek az optimalizáció folyamán. Nem szabad

figyelmen kívül hagyni azt a tényt, hogy a paramétertér korlátainak kiválasztásával korlátozzuk a D tartományt.

A második pontban az első generáció egyedeit kell meghatározni lehetőleg úgy, hogy az egyes egyedek által kódolt paraméterekkel definiált rendszer elégítse ki az alapvető elvárásokat. Ezen a ponton már kezdetben elkerülhetők instabil viselkedések abban az esetben, ha elegendő előismerettel rendelkezünk a rendszerről. Például ha a költségfüggvény a szabályozás jóságát minősítő hibanégyzet-integrál, és a tranziens kiértékelésekor az integrál egy felső határt már elért, akkor az integrálás folytatásától már el lehet tekinteni, mert az egyed instabil rendszert ábrázol.

Minden optimumkeresés sarkalatos pontja az algoritmus harmadik lépéseként felvázolt jósági tényező (fitness) meghatározásánál használatos optimalizációs szempontok megválasztása. A még nem tanított egyedeket „fiatal" egyednek tekintjük, amelyek mindegyikéhez ki kell számítani a jósági tényezőt. Instabil rendszert ábrázoló egyed hibaintegrálja rögzített nagy érték, jósági tényezője gyakorlatilag nulla. A jósági tényezők meghatározásakor használt kritérium esetleg változhat, a folyamat során kibővíthető az egyes egyedek tanulékonyságának figyelembevételével.

A negyedik pont új az algoritmusban. Ezen a ponton jut kifejezésre az egyedek saját élettapasztalatának megszerzése. Az e pontban definiált tanulás a fiatal egyed paramétereiből kiinduló lokális optimumkereső eljárást takar. Ezen a ponton elvileg bármilyen ismert eljárást használhatunk. E munkában használt eljárás fuzzy rendszert modellező neurális hálózat tanításán alapul. Neurális hálózat alkalmazása dinamikus rendszerek optimalizálásánál minden esetben magában hordozza a lokális optimumban való leragadás vagy körülötte történő oszcilláció veszélyét. A genetikus algoritmus hivatott a rendszert kimozdítani a lokális optimumból és a globális optimumba vezetni. A tanulás folyamata, ami minden egyed saját életútját takarja, párhuzamosan végrehajtható. Olyan stratégiát is választhatunk, amikor csak néhány egyedet rendelünk alá tapasztalatszerzésnek, vagy lehet, hogy a genetikus algoritmusban és a lokális minimumkeresésben résztvevő kritériumfüggvények eltérnek egymástól. Jelöljük γ-val a genotípusok terét, ϕ-vel pedig a fenotípusok terét. Legyen $\delta : \gamma \rightarrow \phi$ a fenotípusba történő leképezés. A negyedik lépés elején tehát el kell végezni a δ leképezést. Az egyed tanulása a fenotípusok terében történik. A lokális optimumkeresés végén el kell végezni a δ^{-1} leképezést. Ily módon olyan fenotípust is kaphatunk, amire nem létezik megfelelő genotípus kód. Ennek oka a kvantálás. Ilyenkor választható az ábrázolható legközelebbi genotípus kód, ha a fenotípusról genotípusra kell visszatérni (lásd *2. ábra*). A visszatérés során keletkezett egyedeket „idős" egyedeknek nevezzük.



2. ábra. Globális (A) és lokális (B) keresés

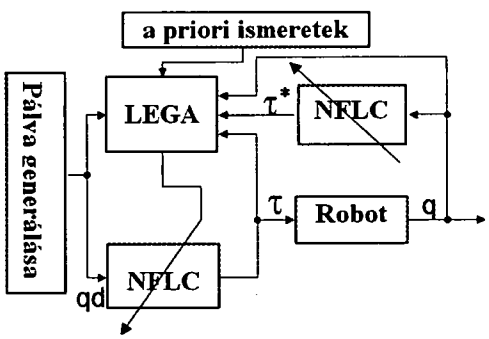
Az ötödik pontra azért van szükség, hogy kiértékelhető legyen a tanulás eredményessége, valamint a további algoritmuspontok már a megváltozott jósági értékekkel működjenek. Új generáció létrehozásához a már jól bevált hagyományos algoritmuselemek használhatók: szelekció, kereszteződés, mutáció, esetleg migráció.

A *2. ábra* mutatja a globális keresés fázisát a genetikus algoritmusban használt kritériumfüggvény által definiált térben (A), valamint a lokális optimumkeresés fázisát a lokális kritériumfüggvénnyel definiált térben (B). A folyamat kiindulópontja a ga és gb pontok (idős egyedek). A genetikus algoritmusban meghatározott operációk eredményeképpen kapjuk a g1 és g2 genotípusokkal meghatározott pontokat (fiatal egyedek). Ezután numerikus optimum kereső eljárással tanítjuk az f1=δ(g1) és f2=δ(g2) fenotípus alakban adott egyedeket, így kapva f1' és f2' -et. Végül inverz transzformációval kapjuk g1' és g2' pontokat (új idős egyedek).

Számos nemlineáris dinamikus rendszer optimalizálása esetében a LEGA algoritmus felhasználásával gyors és sikeres megoldásra jutunk.

3. A LEGA alkalmazása nemlineáris, dinamikus rendszerek esetén

Általános esetben nemlineáris, dinamikus rendszerekre alkalmazható az indirekt tanítási struktúrán alapuló szabályozási rendszer. A 3. ábrán látható tanítás eredményeképp az NFLC-ben (neuro fuzzy logic cotroller) kialakul az irányítandó rendszer (például robot) inverz dinamikus modellje.



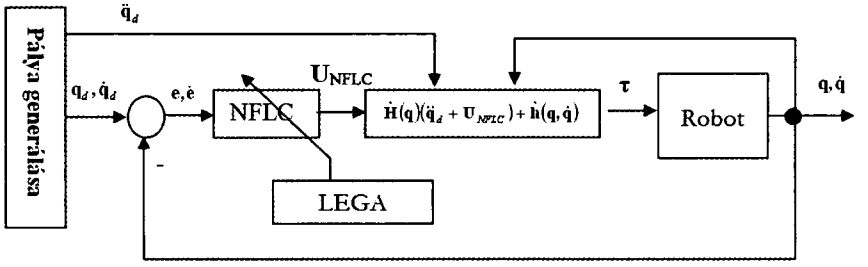
3. ábra. **Robotirányítás NFLC szabályozóval**

Az [5] cikk a dinamikus, nemlineáris rendszerek irányításával kapcsolatban jó áttekintést ad. A LEGA algoritmus alkalmazását a kiszámított nyomatékok módszerénél (Computed Torque Control, CTC) mutatja be a 4. ábra. Tételezzük fel, hogy rendelkezésünkre áll valamilyen előismeret a robotról, és stabil szoft technika felhasználásával meghatározhatunk egy $\hat{\mathbf{H}}(\mathbf{q})\ddot{\mathbf{q}} + \hat{\mathbf{h}}(\mathbf{q},\dot{\mathbf{q}}) = \boldsymbol{\tau}$ nyomatékot. A hagyományos CTC algoritmus a következőképpen definiálja a szabályozási törvényt:

$$\boldsymbol{\tau} = \hat{\mathbf{H}}(\mathbf{q})(\ddot{\mathbf{q}}_d + \mathbf{U}_{PD}) + \hat{\mathbf{h}}(\mathbf{q},\dot{\mathbf{q}}). \qquad (1)$$

Ebben az esetben $\mathbf{U}_{PD}$ helyettesíthető NFLC-vel ($\mathbf{U}_{NFLC}$), aminek következtében neuro-fuzzy kompenzátorral ellátott CTC szabályozót kapunk.

A *4. ábrán* látható architektúrában az NFLC két bemenetű elsőrendű Takagi-Sugeno (TS) féle szabályozókból áll, $e_i, \dot{e}_i$ bemenő jelekkel. Az $\mathbf{e}(t) \in \Re^n$ pályakövetési hiba meghatározható, mint a $\mathbf{q}_d(t) \in \Re^n$ előírt pálya és a $\mathbf{q}(t) \in \Re^n$ realizált pálya különbsége: $\mathbf{e} = \mathbf{q}_d - \mathbf{q}$.



***4. ábra.* CTC szabályozó NFLC kiegészítő szabályozóval**

A rendszer bemenetei felett definiáljunk a fuzzy tagsági függvényeket a következő módon:

$$\mu(x) = \mu_{GBell}(x; a, b, c) = \frac{1}{1 + \left| \frac{x - c}{a} \right|^{2b}} \tag{2}$$

ahol a, b, c szabad paraméterek. Egy $r_{\alpha,i}$ fuzzy szabályt meghatározhatunk a következő módon:

$$r_{\alpha,i}: \; y_{\alpha,i} = \left[\prod_{j=0}^{M_i - 1} \mu\left(e_i^{(j)}, a_{i,j,k}, b_{i,j,k}, c_{i,j,k}\right) \right] f_{\alpha,i}; \quad k = 1,.., L_{i,j} \tag{3}$$

ahol $\alpha = \left(k_{1i}, k_{2i}, \ldots, k_{M_i,i}\right)$; $k_{ji} \in \left\{1, \ldots, L_{i,j}\right\}$. Itt i jelöli a csukló sorszámát, $i = 1, \cdots, N$; α pedig a szabály sorszámát, $\alpha = 1,.., O_i$; $O_i = \prod_{j=0}^{M_i - 1} L_{i,j}$ az adott csuklóban. M_i határozza meg a bemenetek számát az egyes csuklókban, k jelöli az i-edik csukló (j+1)-edik bemenetét lefedő fuzzy tagsági függvény indexét, $L_{i,j}$ pedig az i-edik csukló (j+1)-edik bemenetét lefedő fuzzy tagsági függvények számát. O_i jelöli az egyes csuklókhoz tartozó szabályok számát. Az elsőrendű TS szabályozó következmény (konzekvens) részében szereplő függvények a következők:

$$f_{\alpha,i} = \sum_{j=1}^{M_i} \left(\theta_{\alpha,i,j} \cdot e_i^{(j-1)} \right) + \theta_{\alpha,i,0} \tag{4}$$

Jelölje $w_{\alpha,j} = \left[\prod_{j=0}^{M_i-1} \mu\left(e_i^{(j)}, a_{i,j,k}, b_{i,j,k}, c_{i,j,k}\right)\right]$ az $r_{\alpha,i}$ reláció tüzelési súlyát, akkor a defuzzifikációs eljárás a következő:

$$u_i = \frac{\sum_{\alpha=1}^{O_i} \left(w_{\alpha,i} \cdot f_{\alpha,i}\right)}{\sum_{\alpha=1}^{O_i} w_{\alpha,i}}. \tag{5}$$

Az optimalizáció során **a**,**b**,**c** és θ paraméterek a LEGA algoritmus szerint hangolódnak. Így a hangolandó paraméterek száma a következőképpen alakul:

$$N_\theta = \sum_{i=1}^{N} \left(O_i \cdot (M_i + 1) + 3 \cdot \sum_{j=0}^{M_i-1} L_{i,j} \right) \tag{6}$$

Minden optimalizációnak sarkalatos pontja a kritériumfüggvény meghatározása. Szabályozástechnikában gyakran használatos a követési hiba energiájának minimalizációja:

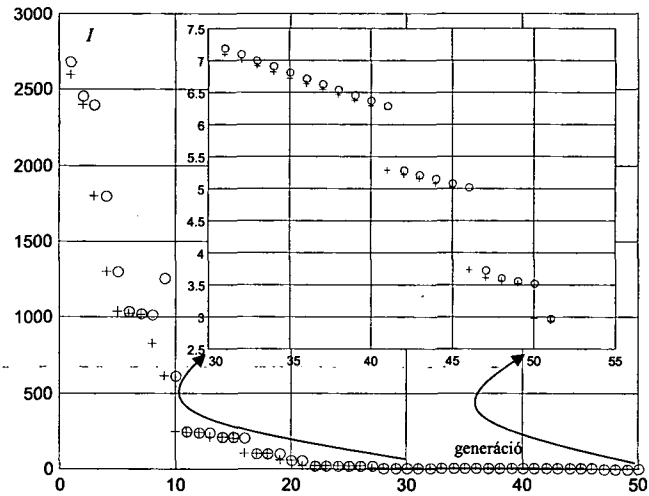
$$I_i = \left(\int_{t_0}^{t_f} \left(k_{1i} \cdot e_i^2\right) dt + \int_{t_0}^{t_f} \left(k_{2i} \cdot \dot{e}_i^2\right) dt \right), \tag{7}$$

a teljes rendszerre vonatkozóan $I = \sum_{i=1}^{N} I_i$. Ha a hiba energiájának minimalizálása mellett büntetni akarjuk a nagy beavatkozásokat, akkor a kritériumfüggvény a következőképpen módosulhat:

$$I_i = \left(\int_{t_0}^{t_f} \left(k_{1i} \cdot e_i^2\right) dt + \int_{t_0}^{t_f} \left(k_{2i} \cdot \dot{e}_i^2\right) dt + \int_{t_0}^{t_f} \left(k_{3i} \cdot |\tau_i|\right) dt \right) \tag{8}$$

Az egyes k_{ij} pozitív súlytényezők meghatározása a szakértő feladata.

Tekintsük a SCARA robot irányítását CTC és NFLC együttes alkalmazásával. Az *5. ábra* nem csak a konvergencia sebesség növekedését mutatja a hagyományos genetikus algoritmussal szemben, hanem szépen illusztrálja a kritériumfüggvény más, meredekebb kontúrok mentén történő változását. Az algoritmus esetében a 30 egyedből álló populáció 4 egyedét ANFIS struktúrával hangoltuk. Az egyes egyedeknek a lokális optimalizáció számára történő kiválasztásánál figyelembe vettük az egyedhez rendelt kritériumfüggvényt. A kiválasztott egyedek felét a legjobb kritériumértékekkel rendelkező egyedek képezték, a maradékot véletlenszerűn választottuk ki. A kiválasztottak esetén a tanulás folyamán új kritériumértékek alakultak ki. Az egyedeket visszavezetve a genotípusokkal definiált térbe, azok már a megváltozott objektív értékekkel térnek vissza a genetikus optimalizálás *1. ábra* szerinti folyamatába. A bemutatott LEGA algoritmusnak igen nagy az erőforrás igénye. Minden egyes egyed esetén el kell végezni a megfelelő nemlineáris dinamikus rendszer szimulálását. Ez a lépés párhuzamosítható. Lehetőség mutatkozik az egyes egyedek által leírt rendszerhez tartozó tagsági függvény egyidejű meghatározására, valamint az ANFIS hangolás is elvégezhető párhuzamosan.



5. ábra. **A legjobb egyed kritériumfüggvényének alakulása LEGA algoritmus alkalmazása esetén, ahol 'o' mutatja a genetikus algoritmus, '+' az ANFIS utáni állapotot generációnként**

4. A LEGA megvalósítása párhuzamos számítással

A javasolt algoritmus erőforrásigénye igen nagy. A végrehajtási idő a feladatvégzések párhuzamosításával csökkenthető. A javasolt számítási architektúra egy mester és több szolga számítási egységet tartalmaz. A genetikus algoritmus elemeinek párhuzamosításával a [1] hivatkozás foglalkozik. Az egyes szolgák által számított egyedek száma (9) szerint alakul.

$$subpop\_size = \frac{pop\_size}{N\_workers} \tag{9}$$

A mester gép az alábbi kódot hajtja végre:

```
1. Az algoritmus paramétereinek inicializálása
2. Az első generáció egyedeinek meghatározása P (populáció).
3. Az N_workers slave folyamatainak létrehozása.
4. for i:=1 to N_workers
5.  for j:=1 to subpop_size
6.    SP[i][j]=P[subpop_size*(i-1)+j]
7.  end for
8.  A subpopuláció elküldése az i-edik slave-nek.
9. end for
10. for i:=1 to n_epoch
11.  for i:=1 to N_workers
12.    SP[i] fogadása a slave folyamatoktól
13.    for j:=1 to subpop_size
14.        P[subpop_size*(i-1)+j]=SP[i][j]
15.    end for
16.  end for
17.  Az egyedek szelekciója.
18.  Az egyedek keresztezése.
19.  Az egyedek mutációja.
20.  A subpopulációk elküldése a slave-eknek.
```

A master gépen futó folyamat az alábbiak szerint zajlik:

1. A paraméterek fogadása a master folyamattól.
2. Az SP[i] (supopuláció) fogadása a master folyamattól.
3. Jósági tényező hozzárendelése az SP[i] egyedekhez. (dinamikus rendszer szim.)
4. A paraméterek transzformálása genotípus térből fenotípus térbe
5. Az egyedek tanítása. (dinamikus rendszer szimuláció és FLC hangolása)
6. A paraméterek transzformálása fenotípus térből genotípus térbe
7. A megváltozott jósági tényezők kiértékelése.
8. Az SP[i] supopuláció és I[i] átadása a master folyamatnak.

5. Összefoglaló

A munkában bemutatásra került LEGA alkalmas a dinamikus nemlineáris rendszerek hatékony optimalizálására. A dolgozat rávilágít az algoritmus erőforrásigényére, de egyúttal megoldást ad annak párhozamosítására. A dolgozatban bemutatott példa illusztrálja a javasolt algoritmus hatékonyságát.

Irodalomjegyzék

Sven E. Eklund (2004): "A massively parallel architecture for distributed genetic algorithms", Parallel Computing 30, pp. 647–676.

N. Nariman-Zadeh, A. Darvizeh, M.H. Dadfarmai (2004): "Design of ANFIS networks using hybrid genetic and SVD methods for the modelling of explosive cutting process", Journal of Materials Processing Technology 155–156, 2004, pp. 1415–1421.

O. Kaynak, I. Rudas (1995): "Application of Soft Computing Methodologies in Mechatronics", Proceedings of the First Asian Control Conference, pp. 53–56, Tokyo, Japan.

Gy. Mester, Szilveszter Pletl (1996) "GANFIS Control Algorithm of Intelligent Robots", Second ECPD International Conference on Advanced Robotics, Intelligent Automation and Active Systems, 155–160.

B. Lantos: "Some Applications of Soft Computing Methods in System Modeling and Control", Procedings of IEEE International Conference INES'97, Budapest, 469–474.