SLOVENSKÁ TECHNICKÁ
UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY
A INFORMATIKY

Ing. Branislav Kadlic

Autoreferát dizertačnej práce

NÁVRH EVOLUČNÝCH ALGORITMOV PRE RIADENIE PROCESOV

na získanie akademickej hodnosti doktor (philosophiae doctor, PhD.)

v doktorandskom študijnom programe: **Kybernetika**
v študijnom odbore 9.2.7. Kybernetika

Miesto a dátum: Bratislava, 10.08.2016

SLOVENSKÁ TECHNICKÁ UNIVERZITA
V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Ing. Branislav Kadlic

Autoreferát dizertačnej práce

NÁVRH EVOLUČNÝCH ALGORITMOV PRE RIADENIE PROCESOV

na získanie akademickej hodnosti doktor (philosophiae doctor, PhD.)

v doktorandskom študijnom programe:
Kybernetika

Miesto a dátum: Bratislava, 10.08.2016

Dizertačná práca bola vypracovaná v externej forme doktorandského štúdia

Na Ústav robotiky a kybernetiky, Fakulta elektrotechniky a informatiky, Slovenská technická univerzita v Bratislave

Predkladateľ: Ing. Branislav Kadlic
Fakulta elektrotechniky a informatiky STU
Ilkovičova 3, 812 19, Bratislava, Slovenská Republika

Školiteľ: doc. Ing. Ivan Sekaj, PhD.
Fakulta elektrotechniky a informatiky STU
Ilkovičova 3, 812 19, Bratislava, Slovenská Republika

Oponenti: prof. Ing. Alajos Mészáros, PhD.
Fakulta chemickej a potravinarskej technologie STU v Bratislave
Radlinského 9, 812 37 Bratislava, Slovenská Republika

prof. Ing. Juraj Spalek, PhD.
Elektrotechnická fakulta Žilinská univerzita
Univerzitná 1, 010 26 Žilina, Slovenská Republika

Autoreferát bol rozoslaný: 10.08.2016

Obhajoba dizertačnej práce sa koná: 24.08.2016 o 10:00 h.

Na Ústav robotiky a kybernetiky, FEI STU v Bratislave
Ilkovičova 3, 812 19 Bratislava, Slovenská Republika, miestnosť D424

prof. Dr. Ing. Miloš Oravec
dekan FEI STU v Bratislave

Obsah

1 Úvod

Evolučné algoritmy (EA) [22,36,28,57] sú efektívny nástroj riešenia širokého spektra praktických problémov, ktoré sa dajú formulovať ako optimalizačné alebo prehľadávacie úlohy. Ich potenciál sa dá úspešne využiť v oblasti návrhu regulačných obvodov a algoritmov riadenia. Takéto aplikácie môžeme z istého pohľadu rozdeliť na dve triedy. Prvá trieda aplikácií je charakterizovaná tým, že štruktúra regulačného obvodu alebo riadiaceho algoritmu je vopred známa a pevne definovaná. To znamená, že vnútorná štruktúra regulátora alebo riadiacej jednotky je určená a objektom návrhu alebo optimalizácie je hľadanie hodnôt vopred známeho a nemeniaceho sa počtu parametrov. Na takto definovanú úlohu sa spomedzi EA zvyčajne používajú prístupy ako Genetické algoritmy (GA), Evolučné stratégie (ES), Diferenciálna evolúcia (DE), Kŕdľové algoritmy (PSO), Umelý imunitný systém (UIS), prípadne iné numerické optimalizačné metódy [7,28,57,64]. Do druhej triedy aplikácií patria také problémy, kde okrem hodnôt parametrov nie je známy ani ich počet. Čiže nie je definovaná vnútorná štruktúra, vzájomné prepojenie stavených blokov a ani ich počet v riadiacej jednotke či v regulátore. Na takéto problémy je orientovaná aj táto práca. Takéto úlohy sú riešiteľné pomocou prístupov, ktoré majú nástroje na vytváranie a modifikáciu štrukturálnych väzieb optimalizovaných objektov, ako aj hľadanie parametrov objektov s novovzniknutými štruktúrami. Medzi nástroje schopné hľadať a tvoriť nové štruktúry patria prístupy ako Genetické programovanie, Gramatická evolúcia, Karteziánske genetické programovanie a niektoré iné [22,36,61,64]. Genetické programovanie a Gramatická evolúcia sú pomerne všeobecné prístupy s veľkým potenciálom. Štruktúry, ktoré tieto prístupy sú schopné generovať nemajú spravidla žiadne obmedzenia. To znamená, že definujeme iba elementárne stavené bloky, ktoré sú medzi sebou spájané väzbami s minimálnymi obmedzeniami. Takto môžu vznikať rôznorodé objekty, často aj neintuitívne, ako by ich „prirodzene inteligentný“ návrhár nikdy nekonštruoval. Ich výsledky môžu mať nečakané, až prekvapivé vlastnosti. Na druhej strane nevýhodou takýchto „umelo inteligentných“ prístupov je vysoká výpočtová (časová) náročnosť. Prehľadávaný priestor je totiž obrovský a obmedzení v spájaní stavených kameňov je málo. V prípade ich použitia v kybernetických aplikáciách výpočtový čas narastá tak výrazne, že to predstavuje zásadný problém. Iba pre ilustráciu môžeme uviesť, že kým parametrizácia PID regulátora pomocou genetického algoritmu je otázka sekúnd

až desiatok sekúnd, návrh štruktúry SISO regulátora pomocou genetického programovania môže trvať hodiny až dni. Z tohto dôvodu je potrebné hľadať cesty a zjednodušenia, ktoré sú na úkor príliš veľkorysej všeobecnosti schopné znížiť výpočtovú náročnosť. Jedným z úspešných prístupov, ktorý bol využitý v iných aplikáciách je Karteziánske programovanie (alebo Karteziánske genetické programovanie, KGP) [61], ktorý je použitý aj v tejto práci. Jeho podstatou je, že nárast nových štruktúr a prepojení elementárnych stavených blokov, ktorý pôvodne nebol nijak limitovaný sa tu redukuje do štruktúry, kde stavebné bloky sú jednak lokalizované v pevnej pravouhlej (karteziánskej) mriežke, ktorá je naviac vopred obmedzená zvolenou veľkosťou. Algoritmus hľadá typ stavebných blokov, prepojenia medzi nimi a hodnoty ich parametrov. Cieľom tejto práce je prezentovať tento prístup pri návrhu resp. optimalizácii riadenia dynamických systémov, predovšetkým pri návrhu štruktúry regulačných obvodov resp. algoritmov regulácie.

2 Evolučné algoritmy v riadení

Metódy pre návrh regulátorov musia rešpektovať rôzne požiadavky kladené na výkon riadiacich systémov. Regulátory často obsahujú mnoho parametrov a obmedzení. Optimalizačný proces môže byť komplikovaný, nespojitý alebo nekonvexný a často nie je možné analytickými metódami dosiahnuť uspokojivé výsledky. Ako už bolo spomenuté, optimalizačné techniky sú schopné tiež tvoriť nové pravidlá riadenia a neintuitívne riešenia. V nedávnej dobe boli evolučné algoritmy aplikované v riadení procesov na vyriešenie širokého spektra optimalizačných problémov. Táto kapitola sa zameriava na návrh regulátorov pre spojité systémy.

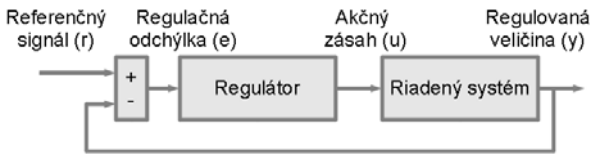
Zhrňme si doposiaľ známe postupy využívajúce evolučné algoritmy do dvoch skupín. Prvá skupina využíva evolučné algoritmy ako výkonnú optimalizačnú stratégiu pre analyticky formulované metódy návrhu riadenia. Parametre regulátorov alebo rôznych dynamických systémov sú navrhnuté analyticky s ohľadom na stabilitu alebo požadovaný výkon systémov [20, 26, 33, 55]. Druhá skupina využíva vyhodnotenie výsledkov simulácie modelu uzavretého regulačného obvodu [13, 21, 31, 39, 62, 63]. Objavuje sa aj prístup optimalizácie viacerých kritérií naraz, kde je medzi kritéria zahrnutá analytická formulácia ale aj výsledok časovej odozvy [63].

Ak návrh zahŕňa nielen hľadanie parametrov vopred pripravenej štruktúry riadenia ale aj hľadanie štruktúry samotnej, je možné použiť genetické programovanie [25, 3, 23, 57, 46]. Pre optimalizáciu štruktúry riadenia môže byť použitý aj hierarchický genetický algoritmus [33]. Genetické programovanie bolo použité ku generovaniu Lyapunovových funkcií [11]. Prehľad návrhov riadení založených na evolučných princípoch [8, 31].

V práci je priamo začlenený prístup založený na vyhodnocovaní výsledkov simulácií časovej odozvy v uzavretej regulačnej slučke. Predstavený prístup sa zaoberá priamym použitím optimalizácie založenej na evolučných stratégiách v hľadaní v priestore parametrov regulátora s rozsiahlymi počítačovými simuláciami navrhnutých systémov v uzavretej regulačnej slučke [60, 59, 57, 46]. Výsledky simulácií sú zásadnou časťou minimalizovanej účelovej funkcie. Neskôr bude ukázané ako je pomocou tohto prístupu návrh optimálnych parametrov pre dynamické systémy pretvorený na štandardný viacrozmerný optimalizačný problém.

2.1 Princíp návrhu regulátora

Ako už bolo spomenuté, cieľom návrhu riadenia je poskytnutie požadovaných statických a dynamických vlastností riadeného procesu, zväčša vo forme dobre známych charakteristických vlastností: maximálne preregulovanie, trvalá regulačná odchýlka, doba nábehu, doba ustálenia alebo iné integrálne ukazovatele kvality [5, 27, ...].



Obrázok 2.1.1: Jednoduchý spätnoväzobný regulačný obvod

S ohľadom na všestrannosť si predstavme jednoduchú regulačnú slučku (Obr. 2.1.1), kde *y* je riadená veličina *u* je riadiaci zásah, *r* je referenčný signál a *e* je regulačná odchýlka (*e*=*r*-*y*). Uvažujme, že je model riadeného systému známy. Kvalita riadenia v uzavretej regulačnej slučke bude vyhodnotená pomocou jednoduchého integrálneho kritéria (2.1.1),

$$I_{AE} = \int_0^T |e(t)|dt \qquad \textbf{(2.1.1)}$$

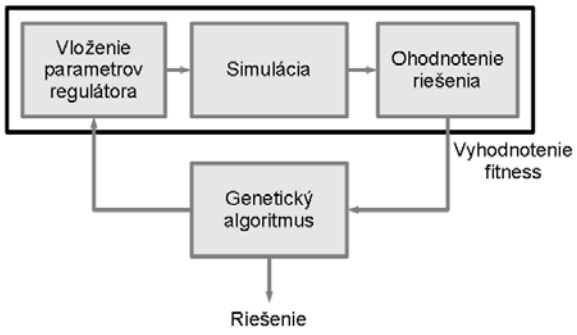
kde T je čas simulácie. Diskrétna forma tohto kritéria kvality (2.1.2),

$$I_{AE} = \sum_{k=1}^{N} T_{s,k}|e_k| \qquad \textbf{(2.1.2)}$$

kde $T_{s,k}$ je krok simulácie a N je počet krokov simulácie.

Návrh regulátora je v podstate optimalizačná úloha. Hľadanie parametrov regulátora z vopred definovaného priestoru parametrov sa uskutočňuje pomocou genetických algoritmov, ktoré sa snažia minimalizovať účelovú funkciu (2.1.1, 2.1.2). Vyhodnotenie účelovej funkcie pozostáva z dvoch krokov. Prvým krokom je simulácia časovej odozvy regulačnej slučky. Druhým je vyhodnotenie účelovej funkcie. Pri návrhu systémov s viacerými vstupmi a viacerými výstupmi (MIMO) alebo inými typmi regulátorov (fuzzy regulátory, regulátory na bázy neurónových sietí) je rozmer priestoru hľadaných parametrov pomerne veľký.

Bloková schéma popisujúca princíp návrhu regulátora za pomoci evolučných algoritmov je znázornená na Obr. 2.1.2.



Obrázok 2.1.2: Bloková schéma návrhu regulátora pomocou evolučných algoritmov

Návrh regulátora pre komplexné systémy môže mať viacero problémov, ako existencia viacerých riešení, či trvanie optimalizačného procesu, kde je často uspokojivé aj riešenie blízke optimu. Prístup k rýchlosti konvergencie návrhu riadenia je podobný ako v prípade iných numerických problémov, v ktorých sa využívajú genetické algoritmy. Závisí od priestoru parametrov, rozmerov prehľadávaného priestoru a výkonu genetického algoritmu.

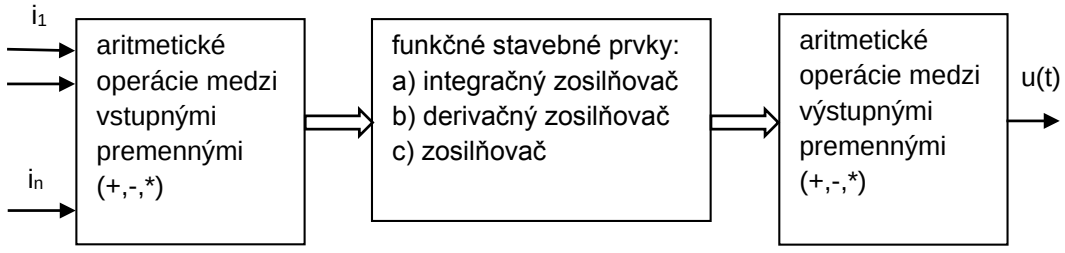
3 Karteziánske genetické programovanie pre návrh regulačných obvodov

Táto kapitola vysvetľuje princíp nášho návrhu aplikácie karteziánskeho genetického programovania pre návrh regulátorov.

3.1 Formulácia problému

Cieľom je aplikácia karteziánskeho genetického programovania pri návrhu regulátorov vo všeobecnosti pre nelineárne dynamické systémy, ktoré sú vytvárané vyberaním a spájaním elementárnych stavebných blokov, ich vzájomných prepojení a hľadaním ich parametrov. Každý regulátor pozostáva z elementárnych jednotiek: blokov a prepojení. Viaceré bloky rôzneho druhu sú prepájané navzájom s cieľom realizovať štruktúru, ktorá generuje riadiacu veličinu pre určený objekt riadenia.

Vstupy blokov sú spracované použitím aritmetických operácií (OP): súčet, rozdiel, násobenie, ktoré definujú matematické vzťahy medzi všetkými vstupnými signálmi konkrétneho bloku. Následne je signál spracovaný dynamickým operátorom, ako: integrátor, derivátor či jednotkové zosilnenie a na konci je násobený zosilnením. Takéto stavebné bloky sú organizované do stĺpcových vektorov, kde výstup z jedného bloku môže byť spojený s ľubovoľným vstupom iného alebo iných blokov. Vstupné signály regulátora (ako napríklad regulačná odchýlka, riadená veličina, merateľná porucha a iné) môžu byť spojené so vstupom každého iného stavebného bloku. Ľubovoľne výstupy blokov môžu reprezentovať výstup(y) regulátora. Takto vytvorená sieť môže byť považovaná za ortogonálnu (karteziánsku) mriežku navzájom prepojených stavebných blokov, kde každý uzol mriežky obsahuje



Obrázok 3.1.1: Jeden základný stavebný blok

elementárnu operáciu. Prepojenie uzlov nie je úplne ľubovoľné. Je limitované vyššie uvedenými pravidlami. Cieľom návrhu regulátora je nájsť optimálnu alebo aspoň vyhovujúcu sieť stavebných blokov, ktoré maximalizujú/minimalizujú zvolenú kriteriálnu funkciu v definovanej regulačnej slučke. Sieť môže byť chápaná ako nelineárna funkcia (3.1.1),

$$u(t)=F(\theta)$$

$$\theta=\{i_1,i_2,...,i_n\};\ i_i \in \{e, e', e'', y, y', y'', s_1 \ldots s_m, d_1 \ldots d_k\} \quad \textbf{(3.1.1)}$$

kde vektor θ obsahuje všetky uvažované vstupné premenné, $u(t)$ je riadiaca veličina (akčný zásah) regulátora, i_i sú vstupné signály regulátora e, e', e'' je regulačná odchýlka a jej prvá a druhá derivácia, y, y', y'' je výstupná veličina a jej prvá a druhá derivácia, s_i sú vnútorné stavové premenné riadeného systému a d_i sú vonkajšie poruchy alebo informácie z prostredia. Je možné použiť akékoľvek signály, ktoré sú k dispozícii.

3.2 Základné stavebné jednotky regulátorov

Jedinec v karteziánskom genetickom programovaní (KGP) je potenciálne riešenie, ktoré je realizované ako funkcia (3.1.1) a reprezentuje kompletnú informáciu regulátora vrátane jej vnútornej štruktúry a parametrov. Každý jedinec je členom populácie. Populácia obsahuje množinu jedincov a reprezentuje základnú dátovú štruktúru karteziánskeho genetického programovania.

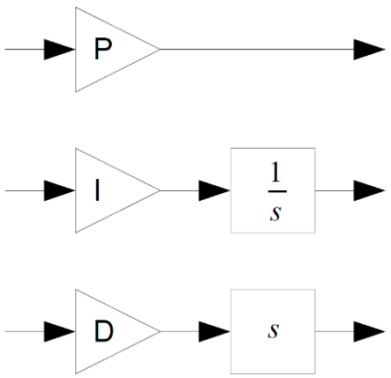
Každý jedinec obsahuje základné stavebné prvky: bloky a prepojenia.

Bloky - elementárne funkčné prvky

Sú parametrizovateľnými prvkami, kde sa vykonáva časť najdôležitejších matematických operácií so vstupnými dátami, pozostávajú z nasledujúcich matematických operácii (Obr. 3.2.1):

- **zosilnenie** - má jeden parameter, ktorým je veľkosť zosilnenia

- **integrátor** - má jeden parameter, ktorý vyjadruje veľkosť zosilnenia prislúchajúce operácii integrovania
- **derivátor** - má jeden parameter vyjadrujúci veľkosť zosilnenia prislúchajúceho operácii derivovania



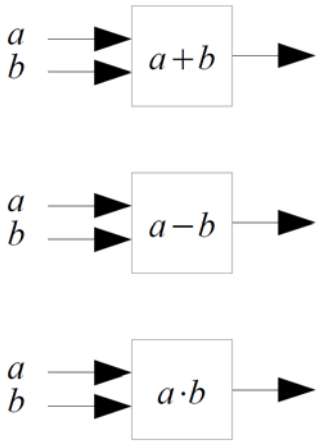
Obrázok 3.2.1: Znázornenie funkčných blokov v laplaceovej transformácií

Prepojenia a matematické operácie

Sú jedným z typov stavebných objektov, ktoré slúžia na vzájomne prepojenie vstupov, výstupov a blokov. Každé prepojenie má typ operácie, s ktorými vychádza zo vstupného bloku, funkčného bloku alebo vstupuje do výstupného bloku, funkčného bloku. Teda prepojenia zabezpečujú správne spojenie rôznorodých zdrojov informácií do jedného tak, aby bol k dispozícii pre blok, do ktorého vstupuje. Zoznam ich operácií je nasledovný (Obr. 3.2.2):

- **súčet** – pre prípad jedného alebo prvého prepojenia vstupujúceho do funkčného bloku alebo výstupného bloku sa táto operácia neprejaví, keďže neexistuje prvý sčítanec, teda prebehne sčítanie nuly a hodnoty z bloku, z ktorého toto prepojenie vychádza, pre prípad viacerých vstupov sa pripočíta k hodnote predchádzajúceho prepojenia
- **rozdiel** – pre prípad jedného spojenia vstupujúceho do funkčného bloku alebo výstupného bloku sa táto operácia prejaví tak, že hodnote z bloku, z ktorého vychádza prepojenie zmení hodnotu na opačnú, pre prípad viacerých vstupov sa odpočíta od hodnoty predchádzajúceho prepojenia

- **súčin** – pre prípad jedného alebo prvého prepojenia vstupujúceho do funkčného bloku alebo výstupného bloku sa táto operácia ignoruje, pre viaceré vstupy sa prejaví ako násobenie hodnoty predchádzajúceho prepojenia



Obrázok 3.2.2:
Znázornenie prepojení
a ich operácií

3.3 Reprezentácia jedinca

Ako bolo uvedené, jeden jedinec (ďalej používame pojem reťazec) populácie je jedno potenciálne riešenie regulátora (riadenia). Pozostáva z viacerých prepojených stavebných blokov. Každý stavebný blok pozostáva z elementárnych prvkov podľa Obr. 3.2.2 a matematických operácií medzi vstupmi a výstupmi bloku (Obr. 3.2.3). Z programátorského hľadiska (z hľadiska reprezentácie genómu jedinca v KGP) bol navrhnutý nasledovný spôsob zakódovania jedinca.

Hlavička

Je použitá na popis vektorov tela, pre statickú dĺžku reťazca je konštantná. Obsahuje nasledujúce prvky:

- počet vstupov regulátora (*I*)
- počet výstupov regulátora (*0*)
- maximálny počet blokov (*B*)
- maximálny počet prepojení (*C*)

Hlavička je tvorená 4 parametrami (*{I, O, B, C}*).

Telo

Je tvorené vektorom parametrov, ktoré popisujú jednotlivé bloky a prepojenia.

Pre popis všetkých funkčných blokov sú potrebné dva vektory (*{{B_T}, {B_G}}*), každý z nich má dĺžku *B*:

- operácia funkčného bloku (B_T)
- veľkosť zosilnenia (B_G)

Pre popis všetkých prepojení sú potrebné tri vektory (*{{C_I}, {C_O}, {C_T}}*), každý z nich má dĺžku *C*.

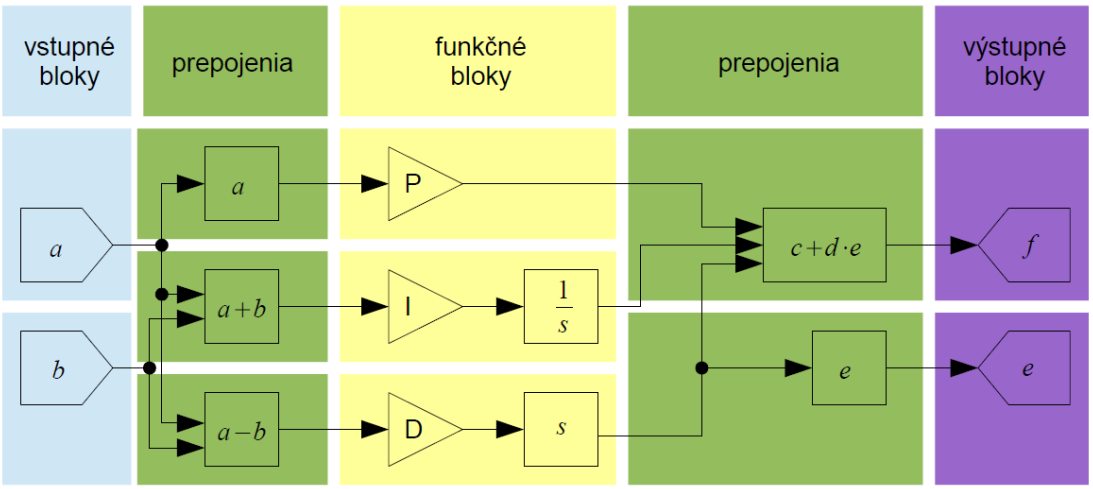
- index bloku, z ktorého prepojenie vychádza (C_I)
- index bloku kam prepojenie vstupuje (C_O)
- operácia, s ktorou hodnoty vstupujú do iného bloku (C_T)

Spolu je teda telo tvorené 5 vektormi *({{B_T}, {B_G}, {C_I}, {C_O}, {C_T}})*.

Všetky vstupné bloky, výstupné bloky a funkčné bloky sú indexované, aby bolo možné jednoznačne definovať prepojenie. Indexácia prebieha automaticky.

- vstupné bloky – sú indexované od *1* po *I*
- funkčné bloky – sú indexované od *1+I* po *B+I*
- výstupné bloky – sú indexované od *1+I+B* po *O+I+B*

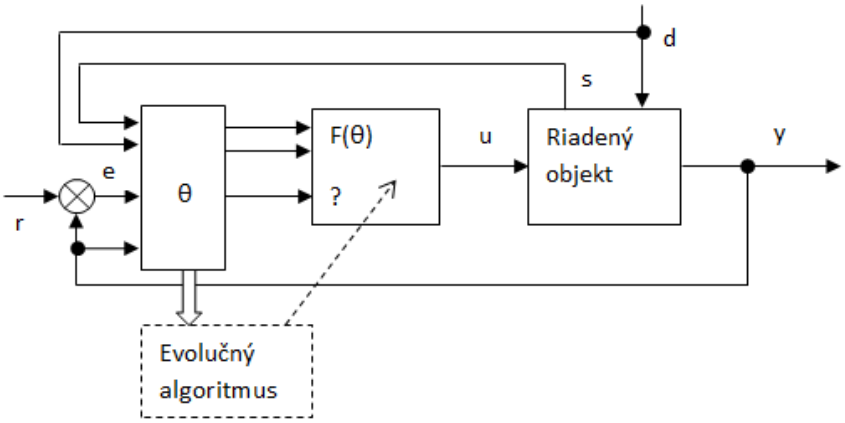
Spolu všetky tieto prvky tvoria reťazec s dĺžkou *D=4+B+B+C+C+C*, ktorý je možné graficky znázorniť (Obr. 3.3.1).



Obrázok 3.3.1: Príklad jedinca (2 vstupné bloky, 2 výstupné bloky, 3 funkčné bloky, 5 prepojení)

Princíp evolúcie riadenia pomocou KGP

Princíp návrhu regulátora, ktorý bol opísaný v predchádzajúcich častiach tejto kapitoly je ilustrovaný na Obr. 3.3.2. Evolučný algoritmus postupne generuje



Obrázok 3.3.2: Schematické znázornenie princípu návrhu riadenia evolučným algoritmom. y – riadené (výstupné) veličiny, u – riadiace veličiny, e – regulačné odchýlky, r – žiadané veličiny, s – vnútorné stavy riadeného objektu, d – externé poruchy resp. informácie o prostredí

náhodné zmeny v populácii jedincov. Jedince, ktoré sú úspešnejšie, majú väčšiu šancu v evolúcii prežiť. Hodnotenie úspešnosti každého jedinca predstavujú dva kroky. Prvý krok je simulácia riadiaceho procesu (uzavretého regulačného obvodu)

vo vhodnom simulačnom nástroji (v našom prípade Simulink). Druhý krok je vyčíslenie zvolenej kriteriálnej funkcie. Tento proces sa opakuje, kým riešenie nevyhovuje, prípadne sa vykonáva stanovený počet generácií.

4 Experimentálne výsledky

V tejto práci navrhnutý a doteraz opísaný prístup návrhu riadenia sme experimentálne overili pri vybraných typoch dynamických systémov. V ďalšom ukážeme výsledky návrhu riadenia nelineárneho dynamického SISO systému, robustifikovaného riadenia, riadenia systému s viacerými vstupmi a výstupmi a riadenia vodnej turbíny.

Pri simuláciách, ktoré sme realizovali, uvádzame ako hlavné výsledky časové priebehy regulovaných veličín, riadiacich veličín, výsledný návrh štruktúry regulátorov a grafy priebehu evolúcie fitness funkcie v závislosti od počtu generácií. Posedne menovaný graf možno považovať za graf priebehu evolúcie návrhu riadenia. Evolučný proces predstavuje minimalizáciu zvolenej kriteriálnej funkcie – kvality regulácie. Pre zosúladenie zo všeobecne zaužívanými postupmi v literatúre, kde sa uvažuje spravidla maximalizácia fitness funkcie (čiže maximalizácia miery úspešnosti) sú ale vo výsledkoch zobrazované hodnoty fitness funkcie maximalizované. To znamená, že fitness funkcia je v našom prípade obrátenou hodnotou minimalizovanej hodnoty kriteriálnej funkcie. Čiže úloha minimalizácie kriteriálnej funkcie (4.0.1),

$$x_{opt}=?;\ f(x_{opt}) = min(f(x)) \qquad \textbf{(4.0.1)}$$

kde x_{opt} je optimum – minimum, $f(x)$ je kriteriálna funkcia, sa prevedie na maximalizačnú úlohu (4.0.2).

$$F(x) = - f(x);\ xopt=?;\ F(xopt) = max(F(x)) \qquad \textbf{(4.0.2)}$$

Teda snažíme sa maximalizovať kriteriálnu funkciu so záporným znamienkom. Z toho dôvodu všetky grafy evolúcie fitness funkcie, ktoré uvádzame v ďalšom narastajú zo záporných hodnôt smerom k nule. Ale nulu nemôžu dosiahnuť, pretože nie je

fyzikálne možné, aby kvalita regulácie (spravidla integrálne kritérium regulačnej odchýlky, atď.) dosiahlo nulovú hodnotu.

4.1 Návrh riadenia vodnej turbíny

Model turbíny a tlakového potrubia je definovaný tromi rovnicami závislými na rýchlosti prúdenia vody v tlakovom potrubí, mechanickým výkonom turbíny a zrýchlením vodného stĺpca. Rýchlosť vody v tlakovom potrubí je (4.1.1),

$$U = k_u G\sqrt{H} \qquad \textbf{(4.1.1)}$$

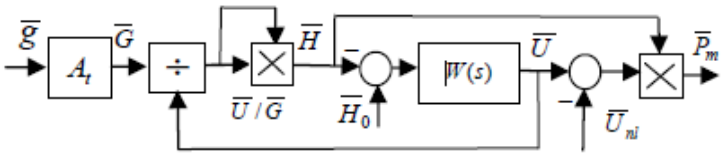
kde U je rýchlosť vody, G je parameter vtoku vody, H je spád hydraulického vtoku, k_u je proporcionálna konštanta. Mechanický výkon turbíny je proporcionálny ku výsledku násobenia tlaku a toku (4.1.2).

$$P_m = k_p HU \qquad \textbf{(4.1.2)}$$

Zrýchlenie vodného stĺpca po zmene v spáde turbíny je popísaný Newtnovým druhým pohybovým zákonom a môže byť vyjadrený v podobe (4.1.3),

$$\rho LA\frac{dU}{dt} = -A\rho a_g(H - H_0), \qquad \textbf{(4.1.3)}$$

kde H_0 je počiatočná ustálená hodnota hodnoty H, A je plocha prierezu potrubia, L je dĺžka potrubia, ρ je hmotnostná hustota, a_g je gravitačné zrýchlenie. Model elektrárne je zobrazený na Obr. 4.1.1 Rovnice (4.1.1), (4.1.2), (4.1.3) sú v normalizovanej forme, kde je uvažovaný nepružný charakter vodného stĺpca.



Obrázok 4.1.1: Bloková schéma hydraulickej turbíny

Ak predpokladáme, nepružný charakter vodného stĺpca bude prevodová funkcia W(s) vo forme (4.1.4),

$$W(s) = \frac{1}{T_w s} \tag{4.1.4}$$

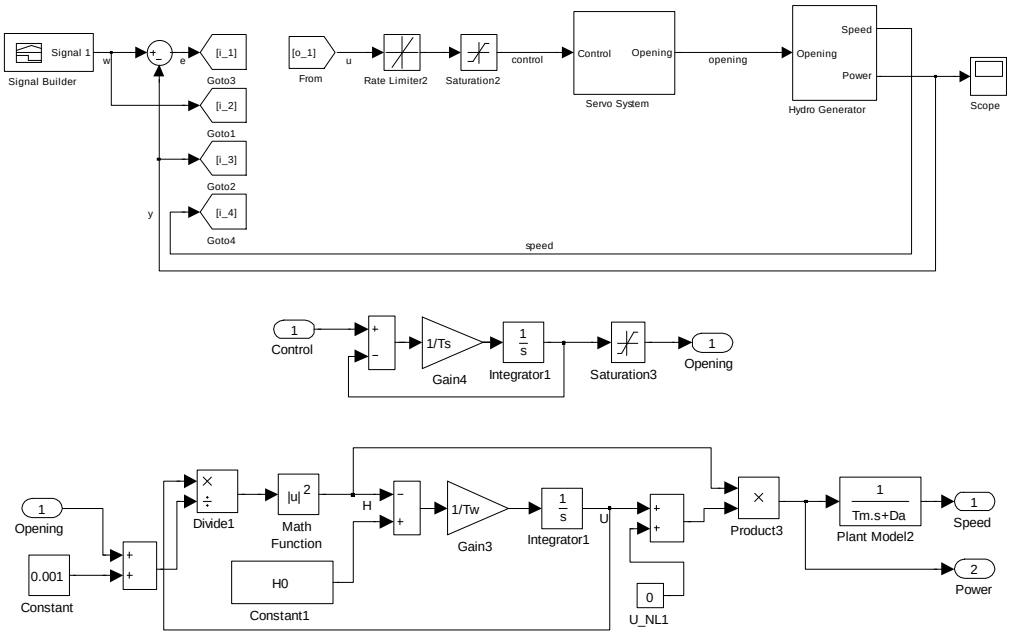
kde T_w je čas nábehu pri záťaži. Má konštantnú hodnotu pre dané potrubie a je stanovená podľa (4.1.5),

$$T_w = \frac{LG_r}{a_g A H_r} \tag{4.1.5}$$

kde H_r a Q_r sú nominálne hodnoty hydraulického spádu a vtoku a respektíve miery toku. Nasledujúce dáta súvisiace s turbínou, potrubím a generátorom sú nasledujúce: dĺžka potrubia *700 m*, miera hydraulického spádu je *180 m*, prierez potrubia je *10,25* m_2 rýchlosť prietoku vody pri nominálnom zaťažení je *75* m^3/s, vtok pri nominálnom zaťažení je *0,96*, vtok bez zaťaženia je *0,04*, výkon turbíny je *150 MW*, nominálny výkon *140 MVA* a časová konštanta nábehu vody pri nominálnom výkone je T_w*=2,9 s.*

Simulačná schéma systému je znázornená na Obr. 4.1.2. Na Obr. 4.1.3 je znázornené porovnanie priebehu regulácie pre PID regulátor a regulátor navrhnutý karteziánskym genetickým programovaním. Priebehy riadiacich veličín sú porovnané na Obr. 4.1.4. Nasledujúce výsledky boli dosiahnuté pri optimalizácii funkcie (4.1.6)

$$J = -\int_0^T |e(t)| + 0.1 \cdot |e'(t)|\, dt \tag{4.1.6}$$



Obrázok 4.1.2: Schéma zapojenia

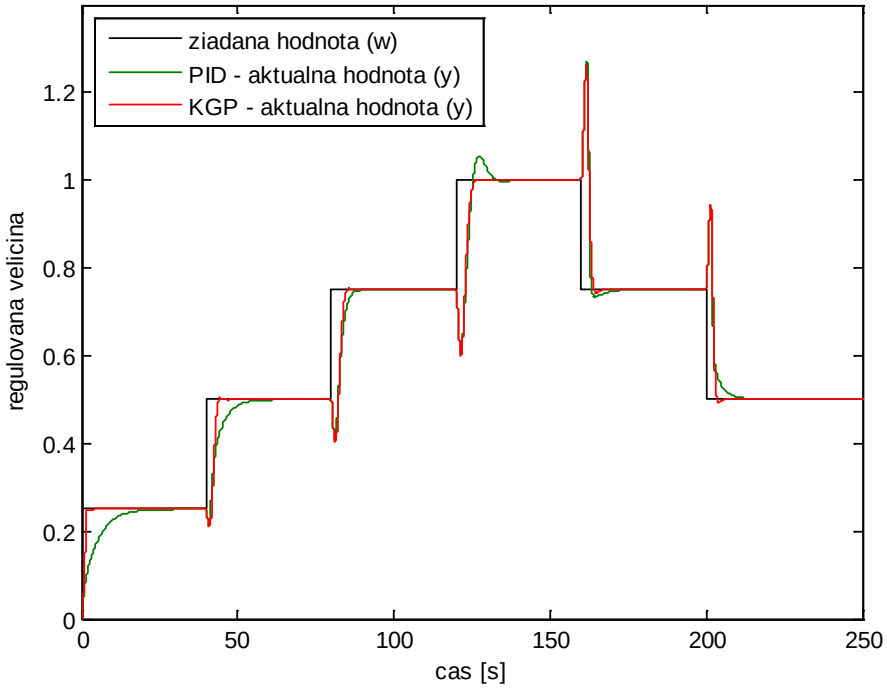
Parameter	Hodnota
P	0,4081
I	0,2430
D	0,1809

Tabuľka 4.1.1: Parametre PID regulátora

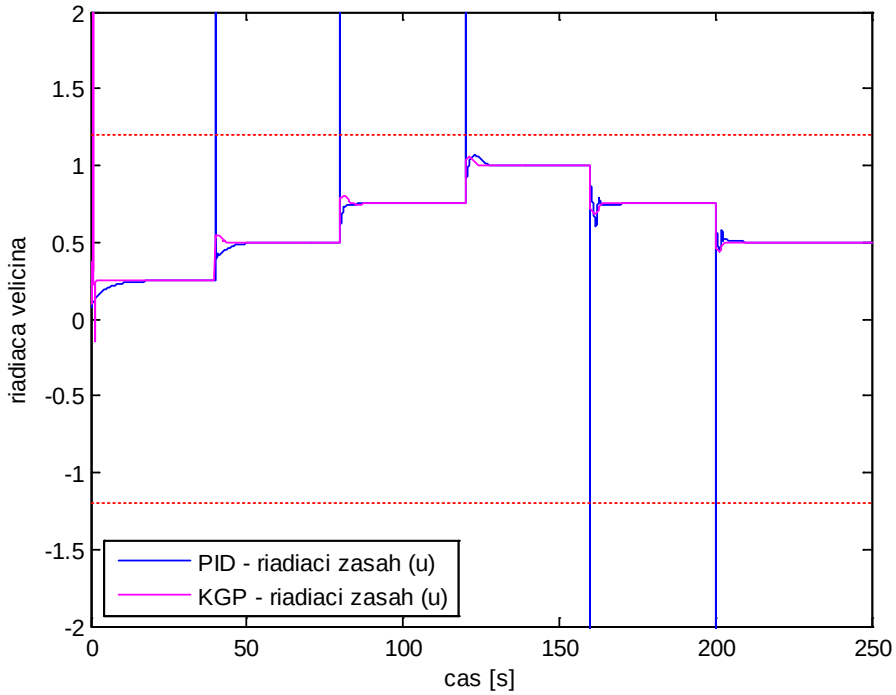
Typ	Fitness
PID	-720.4961
Karteziánske genetické programovanie	-555,7672

Tabuľka 4.1.2: Porovnanie fitness jednotlivých metód

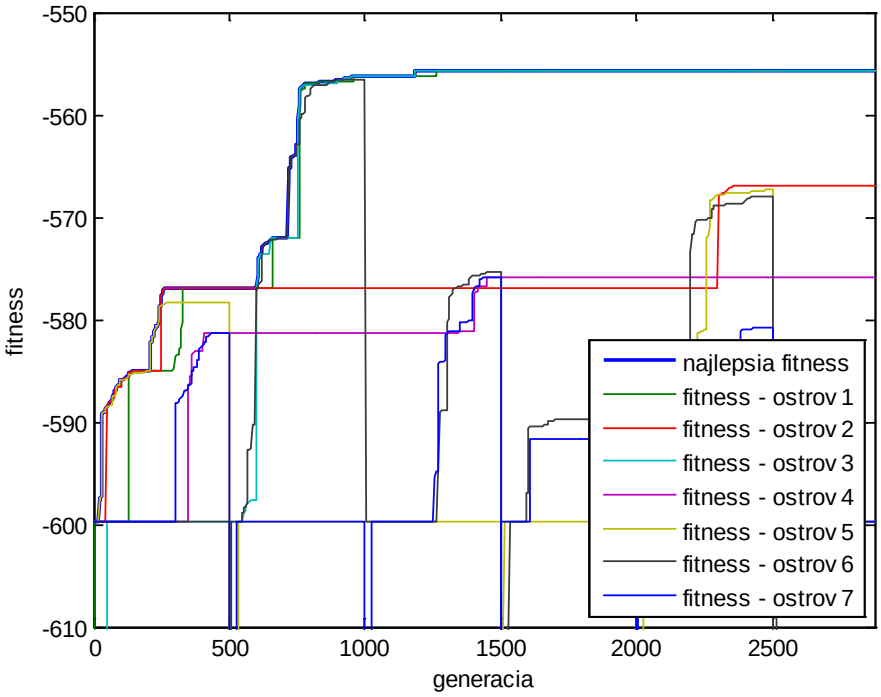
Výsledný regulátor je znázornený na Obr. 4.1.6. Pre systém (Obr. 4.1.2) bolo možné dosiahnuť lepšiu kvalita riadenia, menšie preregulovanie a kratšiu dobu regulácie.



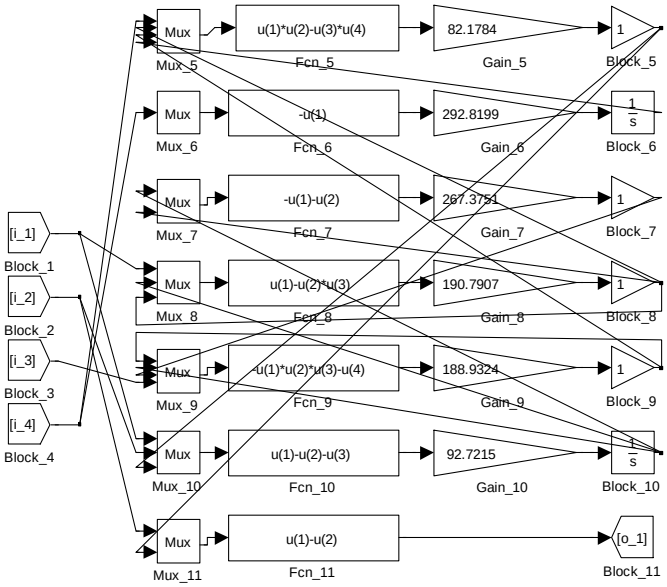
Obrázok 4.1.3: Porovnanie priebehov regulácie nelineárneho systému s PID regulátorom navrhnutým pomocou genetického algoritmu a regulátora navrhnuté pomocou karteziánskeho genetického programovania (KGP)



Obrázok 4.1.4: Porovnanie priebehov riadiacich zásahov PID regulátora a regulátora navrhnutého karteziánskym genetickým programovaním (KGP)



Obrázok 4.1.5: Priebeh fitness pre optimalizáciu regulátora navrhnutého pomocou karteziánskeho genetického programovania (KGP)



Obrázok 4.1.6: Bloková schéma očistená o nevyužívané objekty

5 Záver

V priebehu posledných zhruba dvoch desiatok rokov sa ukázalo, že evolučné algoritmy sú veľmi efektívny optimalizačný prístup, schopný riešiť náročné úlohy z rôznych aplikačných domén. Pritom sa jedná často o úlohy s mnohými optimalizovanými parametrami, s neznámou vnútornou štruktúrou, ktoré sú inými prístupmi riešiteľné iba ťažko, alebo vôbec nie.

Tieto prístupy boli okrem iného úspešne aplikované aj v oblasti riadenia a pri návrhu regulačných obvodov. Medzi úspešných reprezentantov EA v tejto sfére použitia patria predovšetkým Genetické algoritmy. Ako sme už skôr spomenuli, GA sú vhodné na riešenie úloh, keď vnútorná štruktúra problému, v našom prípade regulátora alebo algoritmu riadenia atď. je známa a neznáme sú iba hodnoty ich parametrov. Naproti tomu, keď nepoznáme vnútornú štruktúru navrhovaného/optimalizovaného problému, je potrebné použiť iné evolučné techniky ako sú Genetické programovanie či v niektorých prípadoch aj Gramatická evolúcia. Veľkou prekážkou týchto prístupov je ale veľká (extrémne veľká) výpočtová náročnosť spôsobená veľkou dimenziou prehľadávaného priestoru. Naviac si treba uvedomiť, že vyhodnotenie účelovej funkcie, ktoré je uskutočňované veľmi veľa krát, v prípade návrhu riadenia dynamického systému je v uvažovaných aplikáciách spojené so simuláciou procesu. Toto spôsobuje, že výpočtové časy pri použití napríklad Genetického programovania boli často neprípustne dlhé (desiatky hodín, dni) [46]. Hlavnou motiváciou tejto dizertačnej práce bolo eliminovať tento nedostatok. Vhodným kandidátom sa ukázalo byť Karteziánske genetické programovanie. Tento prístup bol v minulosti úspešne použitý na niektoré iné aplikácie [19]. To dovoľuje redukovať prehľadávaný priestor a tým aj potrebný počet vyhodnotení účelovej funkcie a teda aj výpočtovú náročnosť a výpočtový čas na prijateľnejšie hodnoty. Napriek tomu ponecháva dostatočnú flexibilitu pre nájdenie vhodného riešenia.

Skúsenému kybernetikovi, ktorý si pozrie výsledky regulátorov získané pomocou KGP v tejto práci neunikne fakt, že navrhnuté regulačné štruktúry sú „neintuitívne“. Človek so skúsenosťami v riadení by ich takto nenavrhol. Napriek tomu sú tieto výsledky takmer vždy lepšie, než výsledky získané konvenčnými, bežne zaužívanými regulačnými štruktúrami. Toto je výrazným špecifikom aj Genetického

programovania. Okrem toho, dá sa predpokladať, že ani riešenia nájdené pomocou KGP nie sú globálnymi optimami zvolených kriteriálnych funkcií. Kvalita regulácie, robustnosť, miera stability, prípadne energetické aspekty a iné kvalitatívne ukazovatele sa postupne zlepšujú s počtom výpočtových cyklov (generácií). Nezriedka ale uviaznu v lokálnom extréme, keďže nájsť globálny extrém môže byť ťažká, až nerealizovateľná úloha.

Napriek tomu, výsledky dosiahnuté v tejto práci ukázali, že KGP má potenciál riešiť úlohy z oblasti riadenia pre rôzne typy problémov. Pôvodné výsledky, ktoré tu prezentujeme sú všeobecne opísané v kapitole 4 a experimentálne výsledky sú dokumentované v kapitole 5. Navrhli sme pôvodný prístup využitia KGP pre návrh regulačných obvodov a ukázali sme úspešné riešenie návrhu štruktúry aj parametrov riadenia pre nelineárne dynamické systémy, systémy s rozvetvenými (netriviálnymi) regulačnými obvodmi, ktoré obsahujú viac vstupov aj výstupov. Ukázali sme aj riešenie aspektu robustnosti. Na základe našich skúseností si trúfneme konštatovať, že nami navrhnutý prístup má potenciál riešiť aj iné typy úloh z oblasti kybernetiky, kde štruktúra hľadaného objektu nie je vopred známa. Môže ísť o oblasti identifikácie dynamických systémov, o návrh algoritmov riadenia spojitých, diskrétnych systémov, o logické riadenie o aplikácie v robotike, mechatronike a podobne. Hlavným obmedzením nášho prístupu je výpočtový čas, ktorý býva dlhší, než pri „konvenčných“ metódach návrhu. Na druhej strane obmedzením nie je typ riadeného objektu. Môže sa jednať lineárny/nelineárny systém, systém so zložitou štruktúrou, mnohými vstupmi a výstupmi. Kritériom návrhu môže byť ľubovoľná miera kvality regulácie, obmedzenia energie, veľkosti vstupných a výstupných veličín. Do úvahy môžeme brať existujúce poruchy, režimy prevádzky, šum a podobne. Jedinou podmienkou návrhu je existencia adekvátneho modelu riadeného objektu, ktorého simulácia je súčasťou vyhodnotenia kriteriálnej funkcie.

Použitá literatúra

[1] Aarstad A.S.: Evolving Locomotion for a Quadruped Robot using Cartesian Genetic Programming and Physics Simulation, University of Oslo, 2012

[2] Ashmore L., Miller J.F.: Evolutionary Art with Cartesian Genetic Programming. Department of Informatics, University of Sussex, Falmer, BN1 9QH, Veľká Británia, 2004

[3] Banzhaf W., Nordin P., Keller R. E., Francone F. D.: Genetic Programming: An introduction. Morgan Kaufmann, San Francisco, 1999

[4] Castro L.N., Timmins J.: An Artificial Immune Network for Multimodal Function Optimisation. In Proceedings on Congress on Evolutionary Computation 2002, IEEE Press, pp. 699-704, 2002

[5] Dorf R. C.: Modern Control Systems, Addison-Wesley publishing Company, 5th edition, 1990

[6] Eberhart R. C., Kennedy J.: A new optimizer using particle swarm theory. In Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan. pp. 39-43, 1995

[7] Eiben A. E., Smith J. E.: Introduction to Evolutionary Computing. Springer, 2003

[8] Fleming P. J., Purshouse R. C.: Evolutionary Algorithms in Control System Engineering: A Survey. In Control Engineering Practice, 10(11), 1223, 2002

[9] Garipov E., Puleva T., Haralanova E.: Modeling and simulation of hydraulic turbine power control., Proc. Int. Conf. on Modeling and Simulation (MS'10 Prague), Chech Republic, 2010

[10] Goldberg D. E.: Genetic Algorithms in Search, Optimisation and Machine Learning. Addisson-Wesley, 1989

[11] Grosman B.; Lewin D.R.: Automatic Generation of Lyapunow Functions Using Genetic Programming, In: 16th World Congress of IFAC, Prague, July 3-8, 2005

[12] Harding S., Miller J. F.: Evolution of Robot Controller Using Cartesian Genetic Programming, EuroGP 2005, LNCS 3447:62–73, 2005

[13] Herrero J.M., Blasco X., Martínez M., Salcedo J.V.: Optimal PID Tuning with Genetic Algorithms for Non Linear Process Models. In Proceedings on the 15th World Congress of IFAC, Barcelona, July 21-2, 2002

[14] Hirayama Y., Clarke T., Miller J.F.: Fault Tolerant Control using Cartesian Genetic Programming., University of York, Veľká Británia, 2008

[15] Kadlic B., Sekaj I., Pernecký D.: Design of continuous-time controllers using cartesian genetic programming. In Proceedings of the 19th World Congress of the International Federation of Automatic Control (elektronický zdroj) : IFAC 2014: 19th World Congress of the International Federation of Automatic Control. Cape Town, South Africa, 24-29 August 2014. (s.l.) : IFAC, 2014, 6982-6987. ISBN 978-3-902823-62-5, 2014

[16] Kadlic B., Sekaj I.: Cartesian Genetic Programming Based Controller Design, Conference. MENDEL ’12, Brno, 2012

[17] Kadlic B., Sekaj I.: Controller Design Based on Cartesian Genetic Programming in Matlab, Conference: TECHNICAL COMPUTING PRAGUE 2011. Praha, 2011

[18] Kadlic B., Sekaj I.: Controller Design Based on Cartesian Genetic Programming in Matlab. Conference: ELITECH ’12, Bratislava, 2012

[19] Kadlic B.: Písomná práca k dizertačnej skúške, Slovenská technická univerzita v Bratislave - Fakulta elektrotechniky a informatiky, 2013

[20] Kawabe T., Tagami T., Katayama T.: A Genetic Algorithm based Minimax Optimal Design of Robust I-PD Controller. In UKACC Int. Conference on Control ’96, Conf. Publication No. 427, IEE, pp.436-441, 1996

[21] Khatib W., Silva V., Chipperfield A., Fleming P.: Multidisciplinary Optimisation with Evolutionary Computing for Control Design. In Proceedings on the 14thWorld Congress of IFAC. Beijing, July 5-9, 1999

[22] Koza J. R..: Genetic Programming. Cambridge, MA, MIT Press, 1992

[23] Koza J.R., Yu J., Keane M.A., Mydlowec W.: Evolution of a controller with a free variable using genetic programming. In Proceedings on the European Conference EuroGP 2000. Edinburgh, Scotland, UK, Lecture Notes in Computer Science, Volume 1802. Berlin, Germany: Springer-Verlag, pp. 91–105, ISBN 3-540-67339-3, April 2000

[24] Koza J.R.: Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press, Cambridge Massachusetts, 1994

[25] Koza J.R.: Genetic Programming. Cambridge, MA, MIT Press, 1992

[26] Krohling R.A., Rey J.P.: Design of Optimal Disturbance Rejection PID Controllers Using Genetic Algorithms. InIEEE Trans. On Evolutionary Computation, Vol.5, No.1, 2001

[27] Kuo B.C.: Automatic Control Systems, Prentice-Hall International Editions, 1991

[28] Kvasnička V., Pospíchal J., Tiňo P.: Evolučné algoritmy. Vydavateľstvo STU, Bratislava, 2000

[29] Lazarus C., Hu H.: Using genetic programming to evolve robot behaviours. In Proc. of the 3rd British Workshop on Towards Intelligent Mobile Robots, 2001

[30] Leitner J., Harding S., Förster A., Schmidhuber J.: Mars Terrain Image Classification Using Cartesian Genetic Programming. Dalle Molle Institute for Artificial Intelligence (IDSIA), SUPSI and Universit`a della Svizzera Italiana (USI), Lugano, Švajčiarsko, 2012

[31] Lewin D.R.: Evolutionary Algorithms in Control System Engineering. In Proceedings on the 16th World Congress of IFAC, July 3-8, Prague, 2005

[32] Lipson H., Pollack J. B.: The golem project. In NATURE volume 406. pages 974-978, 2000 (http://demo.cs.brandeis.edu/golem/ (2010-09-29))

[33] Man K.F.; Tang K.S.; Kwong S.: Genetic Algorithms, Concepts and Deign, Springer, 2001

[34] McKay R., Hoai N. Whigham P., Shan Y., O'Neill M.: Grammar-based Genetic Programming: a survey. Genetic Programming and Evolvable Machines, 2010

[35] Michalewicz Z.: Genetic Algorithms + Data Structures = Evolutionary Programs. Springer, 1996

[36] Miller J. F. a kol.: Cartesian Genetic Programming. Springer, 2011 (ISBN: 978-3-642-17309-7, Dept. of Electronics, The University of York, Heslington, YO10 5DD, UK, jfm7@ohm.york.ac.uk, e-ISBN 978-3-642-17310-3, ISSN 1619-7127 Natural Computing Series, DOI 10.1007/978-3-642-17310-3, Library of Congress Control Number: 2011938670

[37] Miller J. F., Thomson P., Fogarty T. C.: Designing electronic circuits using evolutionary algorithms. arithmetic circuits: a case study. (Genetic Algorithms and Evolution Strategies in Engineering and Computer Science), Wiley, 1997

[38] Miller. J.: An empirical study of the e±ciency of learning boolean functions using cartesian genetic programming approach. volume 2 of Proc. of GECCO, page 1135-1142. Morgan Kaufmann, 1999

[39] Mitsukura Y., Yamamoto T., Kaneda M., Fujii K.: Evolutionary Computation in Designing a PID Control System. In Proceedings on the 14thIFAC World Congress, Beijing, 5-9 july, P.R.China, pp. 497-502, 1999

[40] Nidel, M.: Návrh regulačných štruktúr pomocou nástrojov genetického programovania. Diplomová práca, Slovenská Technická Univerzita, Fakulta Elektrotechniky a informatiky, Bratislava, 2005

[41] Nohejl A.: Grammar-based genetic programming. Diplomová práca, Univerzita Karlova v Praze, Praha, 2011 (http://nohejl.name/age/pdf/Adam-Nohejl-2011-Grammar-based-GP.pdf)

[42] Nohejl A.: Grammatical Evolution. Bakalárska práca, Univerzita Karlova v Praze, Praha, 2009 (http://nohejl.name/age/pdf/AGE-Documentation-1.0.2.pdf)

[43] O'Neill M., Ryan C.: Grammatical Evolution. IEEE Transactions on Evolutionary Computation, 2001

[44] O'Neill M., Ryan C.: Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language. Springer, 2003

[45] Páleník, T.: Návrh regulátorov pomocou genetického programovania. Diplomová práca, Slovenská Technická Univerzita, Fakulta Elektrotechniky a informatiky, Bratislava, 2006

[46] Perkácz J.: Automatická syntéza štruktúry riadenia pomocou genetického programovania. Dizertačná práca, FEI STU Bratislava, 2007

[47] Pernecký D., Sekaj I.: Grammatical evolution based controller design. Príspevok: MENDEL '13, Brno, 2013

[48] Puleva T., Garipov E., Ruzhekov G.: Adaptive Power Control Modeling and Simulation of a Hydraulic Turbine, ISAP, 2011

[49] Reynolds C.W.: An evolved, vision-based behavioral model of obstacle avoidance behaviour. In Christopher G. Langton, editor, Artificial Life III, volume 16 of SFI Studies in the Sciences of Complexity. Addison-Wesley, 1993

[50] Ryan, C., Collins, J.J., O'Neill, M.: Grammatical Evolution: Evolving Programs for an Arbitrary Language, EuroGP',98: Proceedings of the First European Workshop on Genetic Programming, vol. 1391, pp.83 -95 1998: Springer-Verlag, 1998

[51] Searson D., Willis M., Montague G.: Chemical Process Controller Design Using Genetic Programming. University of Newcastle, Veľká Británia, 1998

[52] Sekaj I., Perkácz J., Nídel M.: Controller design based on genetic programming. In Proceedings of the Int. conference Mendel'05. Brno, Czech Republic, June 15-17, 2005

[53] Sekaj I., Perkácz J.: Genetic Programming - Based Controller Design. IEEE Congress on Evolutionary Computation, September 25-28, 2007

[54] Sekaj I., Šrámek M.: Robust Controller Design Based on Genetic Algorithms and System Simulation. In Proceedings on the conference CDC-ECC'05, Seville, Spain, December 12-15, 2005

[55] Sekaj I., Veselý V.: Robust output feedback controller design: Genetic Algorithm approach. In IMA Journal of Mathematical Control and Information, 22, pp. 257-263, 2005

[56] Sekaj I.: Control algorithm design based on evolutionary algorithms. In: Chugo, D., Yokota, S. a kol.: Introduction to Modern Robotics. iConcept Press, Hong Kong, 2011 (ISBN 978-0980733068, http://www.iconceptpress.com/books/introduction-to-modern-robotics)

[57] Sekaj I.: Evolučné výpočty. IRIS, Bratislava, 2005

[58] Sekaj I.: Evolutionary Controller design. In: Wellington Pinheiro dos Santos. Evolutionary Computation, In-Teh, Vukovar, Chroatia, 2009 (www.intechopen.com)

[59] Sekaj I.: Genetic Algorithm Based Controller Design. In 2nd IFAC conference Control System Design'03. Bratislava, Slovak Republic, September 7-10, 2003

[60] Sekaj I.: Genetic Algorithm-based Control System Design and System Identification. In Proceedings on the Int. Conference Mendel'99, June 9-12, Brno, Czech Republic, pp.139-144, 1999

[61] Sekanina L. a kol.: Evoluční hardware. Academia, Praha, 2009 (ISBN: 978-80-200-1729-1)

[62] Sweriduk G.D., Menon P.K., Steinberg M.L.: Design of a pilot-activated recovery system using genetic search methods. In:Optimal Synthesis, 1999

[63] Yang Z.Y., Chan C.W., Xue M.S., Luo G.J.: On-line Temperature Control of an Oven Based on Genetic Algorithms. In Proceedings on the 16th World Congress of IFAC, Prague, July 3-8Molina-Cristóbal, A.; Griffin, I.A.; Fleming, P.J.; Owens, D.H. (2005). Multiobjective Controller Design: Optimising

Controller Structure with GA, In: Proceedings on the 16th World Congress of IFAC, July 3-8, Prague, 2005

[64] Zelinka I. a kol.: Evoluční výpočetní techniky, principy a aplikace. Ben, Praha 2009

[65] http://www.cartesiangp.co.uk/

[66] http://www.wikipedia.org/

[67] http://www.genetic-programming.com/

[68] http://link.springer.com/book/10.1007/978-3-642-18609-7/page/1

[69] http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/LukasSekanina.html