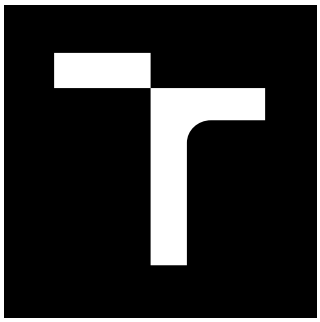



# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

## FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

## ÚSTAV AUTOMATIZACE A INFORMATIKY

INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE


# NELINEÁRNÍ ŘÍZENÍ KOMPLEXNÍCH SOUSTAV S VYUŽITÍM EVOLUČNÍCH PŘÍSTUPŮ

NONLINEAR CONTROL OF COMPLEX SYSTEMS BY UTILIZATION OF EVOLUTIONARY APPROACHES

**DIZERTAČNÍ PRÁCE**
DOCTORAL THESIS


**AUTOR PRÁCE** — **Ing. Petr Minář**
AUTHOR

**ŠKOLITEL** — **doc. Ing. Radomil Matoušek, Ph.D.**
SUPERVISOR



**BRNO 2017**

## ABSTRAKT

Problematika optimalizace složitých soustav za použití algoritmů umělé inteligence, je relativně nový vědní obor a má mnohé způsoby využití v technické praxi. Vhodné algoritmy na řešení podobných úloh jsou třeba genetický algoritmus, diferenciální evoluce, algoritmus HC12, metoda nelder-mead, fuzzy logika a gramatická evoluce. Kompletní řešení je prezentováno na vybraných příkladech od matematických soustav nelineárních systémů, až po praktické úlohy spolu s návrhem antén a stabilizace deterministického chaosu.

Práce si klade za cíl navržení jednotlivých postupů využití algoritmů umělé inteligence při vícekriteriální optimalizaci. K dosažení optimálních výsledků slouží navržené softwarové řešení na základě multi-platformové aplikace v rámci Matlab a Java rozhraní. Softwarové řešení spojuje všechny algoritmy do ucelené aplikace a dále rozšiřuje možnosti uplatnění výsledků na reálných soustavách a v technické praxi.

## KLÍČOVÁ SLOVA

soft computing, umělá inteligence, optimalizace, automatizace, regulace, nelineární systémy, fuzzy logika, chaosové systémy, logistická mapa, duffingova rovnice, uda-yagi, java, matlab, simulink, paralelní výpočty, PID, HC12, DE, NM, GE, GA, TDAS, ETDAS, ITAE

## ABSTRACT

Control theory of complex systems by utilization of artificial intelligent algorithms is relatively new science field and it can be used in many areas of technical practise. Best known algorithms to solved similar tasks are genetic algorithm, differential evolution, HC12 Nelder-Mead method, fuzzy logic and grammatical evolution. Complex solution is presented at selected examples from mathematical nonlinear systems to examples of anthems design and stabilization of deterministic chaos.

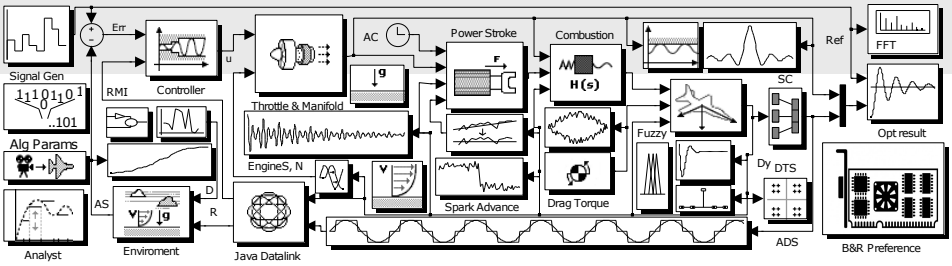
The goal of this thesis is present examples of implementation and utilization of artificial algorithms by multi-objective optimization. To achieve optimal results is used designed software solution by multi-platform application, which used Matlab and Java interfaces. The software solution integrate every algorithms of this thesis to complex solution and it extends possible application of those approaches to real systems and practical world.

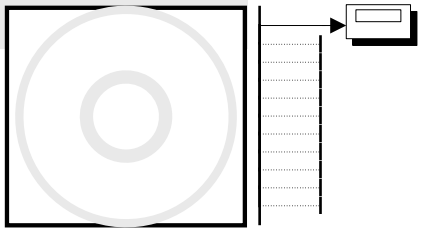
## KEYWORDS

soft computing, artificial intelligence, optimization, automation, control theory, nonlinear systems, fuzzy logic, chaos systems, logistic map, duffing equation, uda-yagi, java, matlab, simulink, parallel computing, PID, HC12, DE, NM, GE, GA, TDAS, ETDAS, ITAE

MINÁŘ, Petr *Nelineární řízení komplexních soustav s využitím evolučních přístupů*: zkrácená dizertační práce. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav procesního a ekologického inženýrství, 2016. 37 s. Vedoucí práce byl doc. Ing. Radomil Matoušek, Ph.D.

# OBSAH

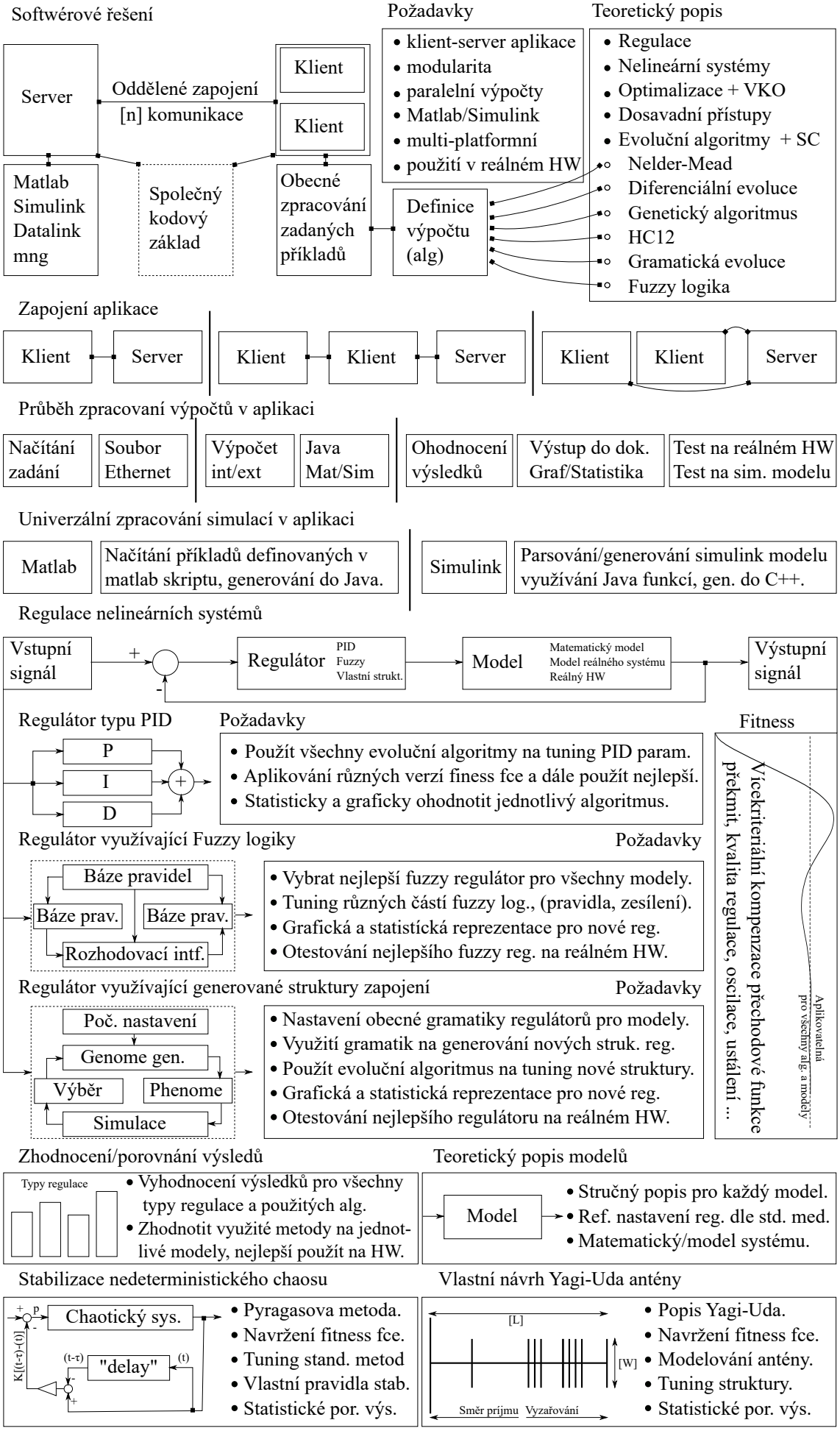


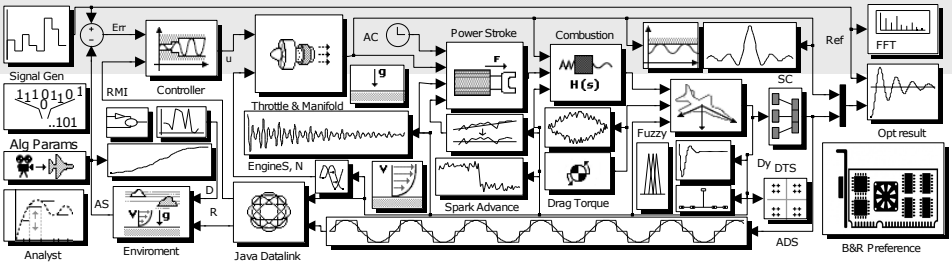
# ÚVOD

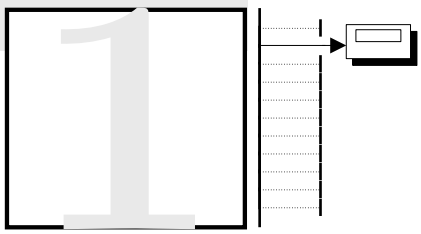
Cílem této disertační práce je efektivní implementace pokročilých optimalizačních metaheuristik, návrh optimalizačních postupu pro vybrané inženýrské úlohy, návrh algoritmu řízení pro uvažované komplexní soustavy a analýza výsledků a zhodnocení řešení. Dále jsou cíle aplikované ve vytvoření univerzální multiplatformní aplikace typu server-klient pro otestovaní, vypočet a vyhodnocení výsledků z oblasti evolučních algoritmů s využitím nejnovějších metod soft computing na teorii řízení a stabilizace chaosu. Práce si klade za cíl prezentovat jednotlivé postupy na jednotlivých typových příkladech. Využití těchto metod je v praxi hojně využíváno a v této práci i odkazováno na různé způsoby řešení, každý z příkladů znázorňuje nové přístupy k této problematice a snaží se ukázat vhodný přístup řešení daného problému. Všechny způsoby jsou prezentovány na základě simulací a testovány na reálné soustavě. Konečné výsledky jsou pak zhodnoceny na základě statistických metod. K dosáhnutí tohoto cíle je zapotřebí napsaní vlastní optimalizační aplikace, která bude aplikovat všechny druhy algoritmů na všech typových příkladech. Aplikace má prezentovat způsob řešení této problematiky v praxi, kde neexistuje žádné univerzální řešení. Dílčí cíle této práce by se daly formulovat takto (obr: 1):

- Přiblížit obecně problematiku regulace nelineárních systémů.
- Obecný popis využívaných evolučních algoritmů a provést rešerši dosavadních postupů.
- Vytvořit a otestovat programové nástroje schopné použít nastíněné postupy řešení, v programovém jazyce Java schopné zpracovat Matlab a Simulink simulace.
- Otestovat a rozšířit postupy při regulaci komplexních nelineárních soustav na předem daných příkladech.
- Otestovat a rozšířit postupy při stabilizaci logistické mapy a duffyngovy rovnice, obecně nedeterministického chaosu.
- Aplikovat poznatky evolučních algoritmů na návrh Yagi-Uda antény.
- Zpracovaní naměřených (simulovaných) hodnot.
- Porovnaní dosazených výsledků proti standardně používaným metodám.




**Obrázek 1:** Ideové schéma a formulace řešení problémů prezentovaných v disertační práci.



# KAPITOLA 1



# OPTIMALIZAČNÍ PŘÍSTUPY A SOFTWAROVÉ ŘEŠENÍ

Obsah pojmů soft computing (SC), počítačová inteligence a umělá inteligence není zcela jednotně chápán. Rovněž popis heuristických metod, či dále optimalizačních metaheuristik ve vztahu k SC, CI, a AI nabízí více pohledů [11]. Soft computing je odvětví počítačových věd, které vzniklo poměrně nedávno a dnes je velmi populární k řešení složitých soustav a komplexních NP problémů. Přístup k (SC) je odvozeno od lidského myšlení a přírodních procesů. Základní idea pro vznik (SC) je odvozena z mnoha zdrojů, jedním z hlavních je [7], [2] a mnoha dalších. Velkou výhodou (SC) metod je jejich jednoduchost a možnost aplikace pro paralelní výpočty. Existuje mnoho metod (SC), ale ne každá se hodí pro všechny druhy problémů, proto je vhodné vzájemně metody kombinovat pro dosažení nejlepšího výsledku. Nejpoužívanější metody (SC) jsou Fuzzy logika (FL), Neuronové sítě (NT), Strojové učení (SU), Evoluční algoritmy (EA) a další [5]. Využití evolučních přístupů v této práci je prezentováno na obrázku (obr: 1.1).

Skupina evolučních algoritmů (EA) jsou stochastické vyhledávací metody využívané k optimalizaci, které se v jádře pokoušejí napodobit proces přírodní evoluce a jsou velice jednoduché k softwarové implementaci. Na počátku (EA) nemá žádné informace o prohledávaném prostoru a závisí zcela na informacích od operátorů (EA), jako je třeba (reprodukce, křížení, mutace). Pomocí těchto a dalších procesů se snaží EA vylepšovat, popřípadě reprodukovat populaci potomků a ve finále najít nejlepší (optimální) výsledek dle zadané hodnotící funkce. (EA) jsou schopné velmi dobře najít optimální výsledky $i$ v $n$ dimensionálním prostoru bez obecných potíží, jako mají třeba gradientní algoritmy (zaseknutí v lokálním minimu). Některé výhody (EA) zahrnují [1]: všechny EA jsou globální optimalizační metody, optimalizace pro spojité i diskrétní parametry, nepotřebují výpočet derivací, pracují dobře s multidimenzionálním prohledáváním, jsou vhodné k paralelním výpočtům, nemají tendenci zaseknutí v lokálním minimu i ve velmi komplexním prohledávacím prostoru, jsou vhodné pro velkou škálu problémů, od analytických funkcí, po experimentální data.

| | | |
|---|---|---|
| Regulátor typu PID | HC12<br>Diferenciální evoluce<br>Nelder-Mead | Testovaní základního nastavení algoritmů na všechny typy hodnoticí funkce. Porovnaní s nejvíce využívaným algoritmem postupem. Popis obsahuje kapitola 3. |
| Regulátor využívající Fuzzy logiky | HC12 | Využití fuzzy logiky na různé typy nastavení regulátoru, jednotný typ fitness funkce. Popis kapitola 4. |
| Regulátor využívající generované struktury zapojení | Gramatická evoluce<br>HC12 - tuning | Generovaní vlastní struktury dle předem definovaných pravidel, využití různých typů gramatik, testování doladění dle vybraného algoritmu. Kapitola 5. |
| Stabilizace nedeterministického chaosu | HC12 - (E)TDAS<br>Gramatická evoluce<br>HC12 - tuning | Hledaní stabilizačních pravidel dle standartních postupů s využitím evolučního přístupu. Generovaní vlastních zákonů a následné doladění. Popis kapitola 7 a 8. |
| Vlastní návrh Yagi-Uda antény | HC12<br>Diferenciální evoluce<br>Nelder-Mead | Hledaní vlastní konstrukce specifických antén s využitím evolučních přístupů pro různé algoritmy. Definice vlastní fitness funkce. Popis kapitola 9. |

**Obrázek 1.1:** Využití evolučních přístupů na různé typy problémů prezentované v této práci a jejich stručná charakteristika.



## 1.1 Popis softwarového řešení

Zdůvodu ověření navrhovaných algoritmů implementovaných majoritně v teorii řízení bylo potřeba vytvořit vlastní optimalizační aplikaci, která obsahuje všechny předešlé algoritmy a je schopná spouštět matematické modely v různých formátech zadání. Zadání modelů je ve formě testovacích příkladů [10] nebo modely k teorii řízení dle syntaxe Matlab skript a Simulink model design a vycházejí z práce [15].

Využití matlab prostředí k řešení optimalizačních úloh je jednoduché a efektivní, implementace vychází z textu [3]. Celkový koncept aplikace vychází z autorových předešlých prací [18] a [4] ze kterých byly odvozené způsoby komunikace mezi výslednými aplikacemi a implementování evolučních algoritmů a jejich škálování v multiprocesorovém prostředí. Dále se z těchto prací používá návrh rozdělení aplikací na *Server* a *Klient* část z důvodu co možná největší univerzálnosti výsledného řešení. Samotná aplikace je napsaná programovacím jazykem Java [8], kvůli její multiplatformosti a celkové robustnosti zvoleného jazyka. Využíté technologie v rámci komunikace jsou TCP/IP [12] a vzdálené volaní funkcí server aplikace [13]. Pro dosažení co nejlepšího výpočetního času na počítačích s více jádrovými procesory se využívá spouštění modelů v rámci technologie Matlabpool [14] v aplikaci Matlab / Simulink a samotné škálování výpočtů v Java aplikaci dle [19] a [6]. Komunikační propojení serverové aplikace a matlab prostředí je provedeno dle technologie Java Matlab Interface [9], který implementuje socket komunikaci mezi hostitelskou Java aplikací a Matlab programem. Implementace evolučních algoritmů a zapojení matlab prostředí k externímu výpočtu problému přebírá některé programové techniky z již existujících prací a to konkrétně [20] a [21]. Výsledky vycházející z běhu optimalizace této aplikace a byly navrženy s co možná největší možností dalšího uplatnění na

reálných soustavách. Jejich podoba podporuje implementaci do programovatelných automatů od firmy B&R Automation, pomocí technologie Target for Simulink [16] a implantování do technologie dSpace [17] v rámci Simulink aplikace.

## 1.2 Obecný popis aplikace

Softwarová implementace je navržená s ohledem na maximální obecnost možné definice optimalizačního problému a volnost definice optimalizačních metaheuristik. Výsledná aplikace je z tohoto důvodu rozdělena na dvě části. Jedna část reprezentuje optimalizační problém a druhá část implementuje optimalizační algoritmy. Schématické rozdělení je na obrázku (obr: 1.2). Optimalizační část (též nazvaná klientská aplikace) zajišťuje jen chod optimalizačního algoritmu a komunikačního protokolu s výpočtovou částí (server aplikace). Toto část vůbec neobsahuje výpočetní prostředky pro řešení daného problému, jen se odkazuje na server a posílá data o optimalizovaných parametrech a server zpátky vrací vyřešený problém, například ve formě fitness funkce. Serverová aplikace je v jádru jen datalinkový program, který zajišťuje komunikaci jak od klientů, tak komunikaci s výpočetní částí (model optimalizačního problému), v našem případě aplikací Matlab / Simulink.



V rámci vývoje výpočetní aplikace se vytvořili i další podpůrné prostředky na analýzu výsledků a podporu řešených problémů. Aplikace pro analýzu problému je část nazvaná jako "Simulation Info generator", její účel je zpracování výpočetních dat a prezentace výsledků na požadovaných hodnotách a výstupech (jak grafická prezentace, tak statistické rozdělení výsledků). V této aplikaci lze i dodatečně analyzovat problém i po ukončeném výpočtu na základě změny parametrů optimalizace a fitness funkce. Další podpůrná aplikace je tzv. "Simulink MDL file parser", jehož účel je zpracování dat z *MDL* souboru, používaném v aplikaci Simulink. Hlavní předností aplikace je využití v rámci algoritmu (GE) a v rámci grafické analýzy jednotlivých Simulink systémů.

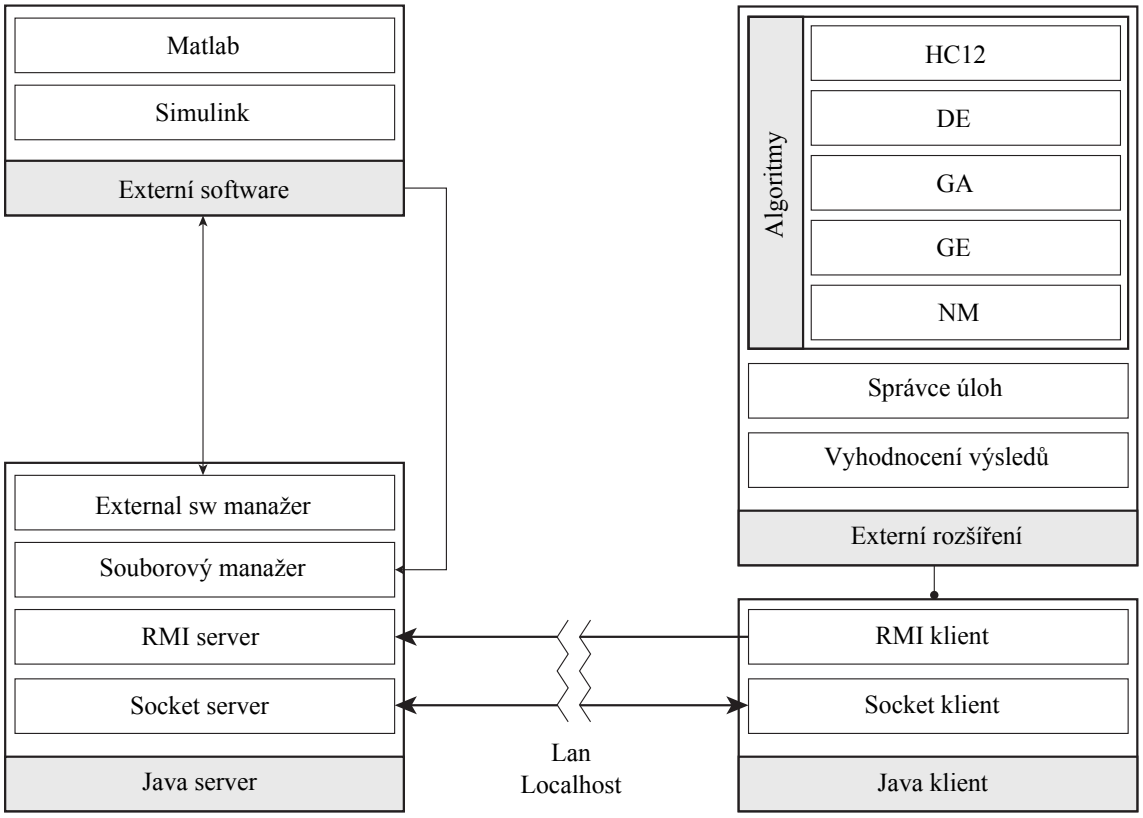
Výhoda uvedeného řešení, kdy server vlastní interní / externí výpočetní část, je schopnost serveru zcela ovládat Matlab a tak využít všech jeho předností a může se zcela soustředit jen na síťovou komunikaci mezi klienty. Další nespornou výhodou je možnost využití distribuovaných výpočtů v režimu $x$ klientů 1 server.

Příklad komunikace, kdy klient chce řešit optimalizaci regulátoru na zadaných parametrech pomocí daného algoritmu. Při inicializaci úlohy klient pošle serveru jen data, co chce řešit a na jakých parametrech. Server si přečte poslaná data a předá je výpočtové části (Matlab nebo Simulink), ta si vytvoří strukturu problému a definuje si návratovou fitness funkci. V průběhu chodu algoritmu klient jen posílá data parametrů serveru, ten je přeposílá výpočtové části a ta vrací hodnotu fitness, která se opět přes server vrací ke klientu. Rozdělení výpočetního problému na podúkoly přináší zlepšení efektivního využití distribuovaných výpočtů a využití multithread procesorů, tedy se jedná o distribuovaný model vypočtu se vsemi přednostmi.

Výhody tohoto návrhu, kdy je řešení rozděleno na tři části, spočívají hlavně v následujících vlastnostech.

- Maximální obecné řešení oddělené od optimalizovaného problému, klient vůbec neřeší, jak se daný problém vypočítá.
- Velice lehká implementace nových algoritmů (vytvoření nového modulu v klient aplikaci).
- Možnost spuštění klientské aplikace na pomalém (obyčejném) PC a připojit se k serveru pomocí lokální sítě nebo internetu. Který může běžet na výkonném výpočetním hardware (výkonné PC, workstation, cluster, superpočítač).
- Jednoduchá implementace paralelních výpočtů, kdy k jednomu serveru se mohou připojit vice klientů.
- Server i klient aplikace jsou napsány v jazyce java, tedy jsou plně mutiplatformní.
- Výpočetní část zahrnuje více vláknový výpočet dle funkce *matlabpool*.






**Obrázek 1.2:** Blokové schéma softwerového řešení (klient/server aplikace).

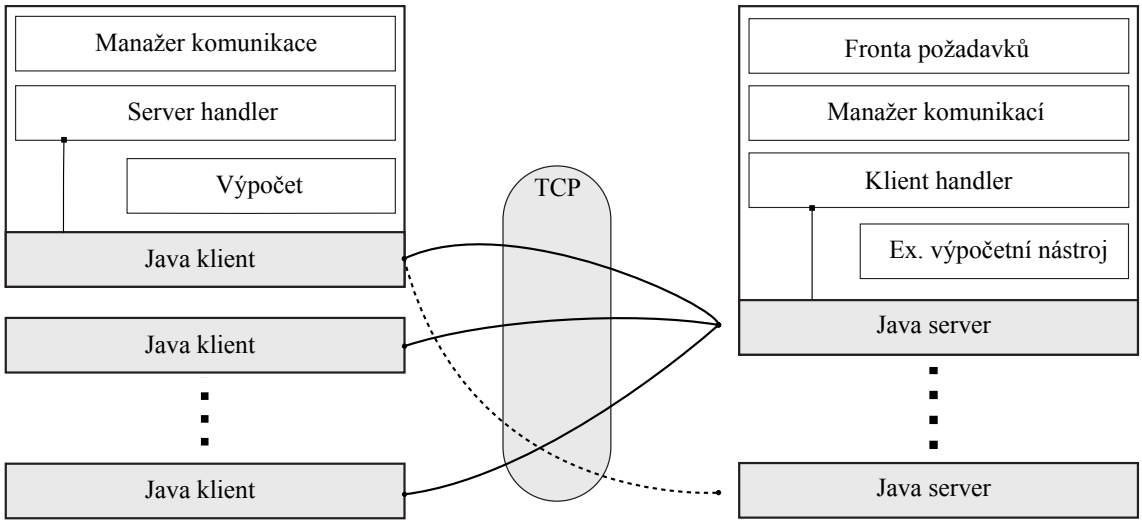
## 1.3 Mezi softwarová komunikace

Způsob komunikace mezi aplikacemi je rozdělen na dva druhy. První je místní komunikace, kdy server zajišťuje spojení mezi výpočtovou částí prostřednictvím protokolu Java Matlab Interface (JMI) a vzdálenou, tedy komunikace mezi serverem a klientem. Tato komunikace je dvojího druhu, první pomocí Remote Method Invocation (RMI), kdy klient volá přímo metody obsažené v serveru. (RMI) komunikace je jednostranná ve směru $klient \rightarrow server$. Poslední využívaný protokol je Socket Interface, pro předávání zpráv mezi klientem a severem v textové podobě, tento typ komunikace je obousměrná $klient \leftrightarrow server$. Vzdálená komunikace lze použít i ve formě localhostu, tedy server i klient jsou spuštěny na jednom PC.

### 1.3.1 Možnosti aplikačního spojení

Mezi klientem a serverem existuje více možností propojení. První základní verze je spojení, ala localhost na jednom PC, kde je jeden klient připojený na jeden server. Tento způsob je vhodný kvůli oddělení výpočetní části s algoritmy od prosté simulace systémů. Další mozností je alokovaní samostatného výkonného PC jen pro server s výpočetní části a na tento server připojený více samostatných klientů. Výhoda tohoto zapojení je, možnost výpočtu v jednom okamžiku několika optimalizačních úloh, které úkolují jen výpočet modelů na externím server PC. Z hlediska času zabere většinu výpočet simulace modelu než samostatný chod algoritmu. Proto je možné spustit klienta třeba na pomalém PC (například notebook) a úkolovat sever, který je nainstalovány na velmi silném PC (workstation). Poslední možností je instalovaní více serverů na více PC a jeden klient připojit na těchto více serverů a při každé nové iteraci rozdělit populaci mezi tyto server počítače. Důvod je urychlení optimalizační úlohy, jak již bylo napsáno, většinu času zabere simulace a výpočet fitness na modelu než samotný chod algoritmu, proto tento způsob poskytuje nejlepší možnost úspory času. Princip jednotlivých zapojení je zobrazen na následujícím obrázku (obr: 1.3).



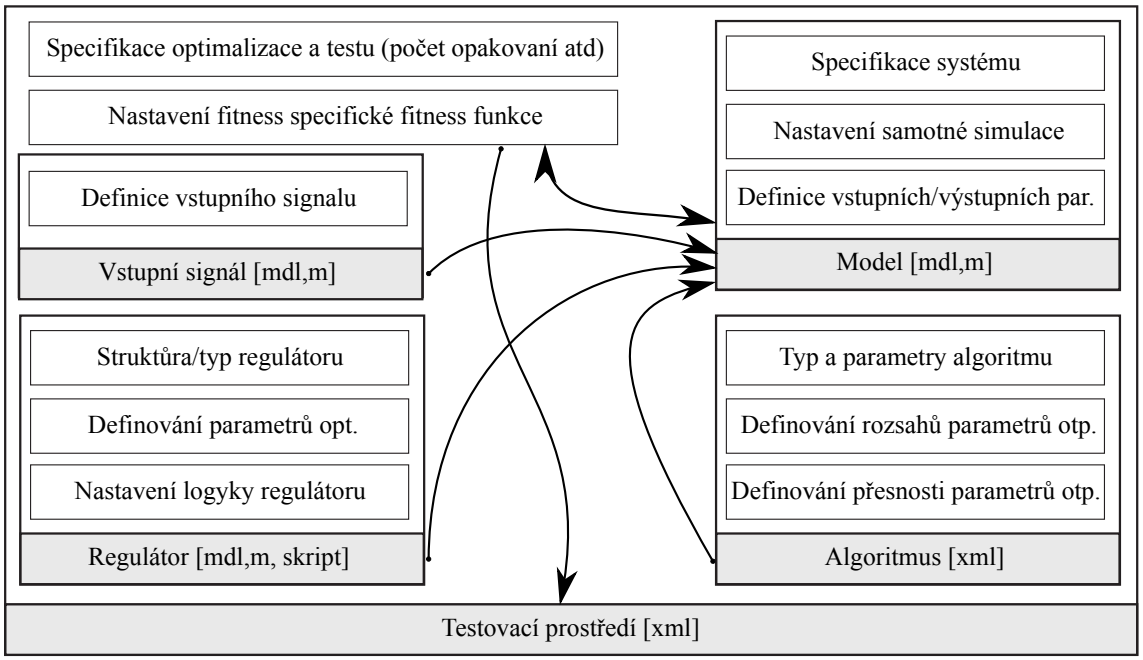



**Obrázek 1.3:** Možnosti aplikačního spojení mezi klientem a serverem.

## 1.4 Nastavení testovacího prostředí

Samotné nastavení optimalizace je uloženo v souboru typu XML. Tento definiční soubor obsahuje přesné specifikace optimalizace a nastavení simulace modelů. Celkové nastavení je rozděleno do pěti oddělených částí. První obsahuje definici optimalizace, konkrétně počet opakování a definici fitness funkce. Druhá část je specifikace modelu s definicí vstupních a výstupních parametrů, definice simulace pro Matlab nebo Simulink a model systému. Třetí část je definice zadaného referenčního signálu pro simulaci. Čtvrtá část je definice modelu regulátoru spolu s parametry optimalizace a logiky regulace. Poslední definice je nastavení parametrů optimalizačního algoritmu, typu algoritmu a nastavení rozsahů parametrů pro optimalizaci s přesností parametrů.

Tento způsob nastavení testovacího prostředí je zvolen kvůli možnosti modulárního spouštění příkladů s rozdílnými definicemi a rozdílnými typy regulátorů. Další výhoda je možnost definování systému, regulátoru a referenčního signálu v různých typech výpočetní definice (*mdl* nebo *m*) a různě je mezi sebou kombinovat.



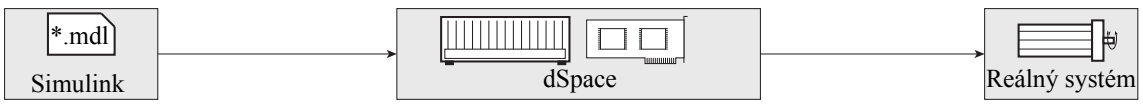

**Obrázek 1.4:** Model nastavení XML specifikace výpočtu.

## 1.5 Možná aplikace v reálných soustavách

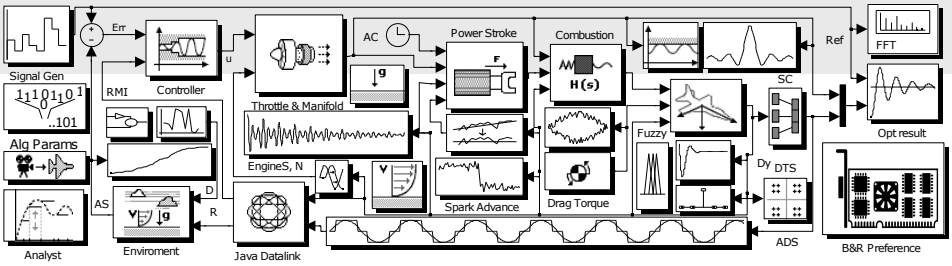
K řízení reálných soustav se využívají embedded systémy s PC, mikrokontroléry nebo programovatelné automaty (PLC). Technologie BR je zvolena také z důvodu lehké aplikovatelnosti výsledku optimalizace na vybrané (PLC) a to generováním kódu (C, C++) přímo z prostředí Simulink a k tomuto účelu se využívá toolbox B&R Automation Studio Target. V této práci je majoritně využíván jako výpočetní nástroj software Simulink, v kterém je i uložen výsledek optimalizace, proto výsledek je lehce aplikovatelný na vybraný automat.

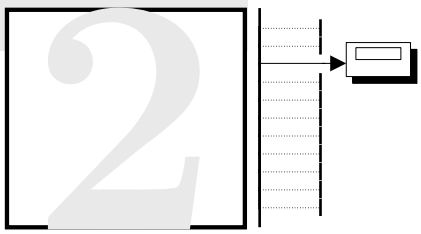

**Obrázek 1.5:** Posloupnost kroků ke generování PLC kódu z prostředí Simulink.

K real-time řízení systémů lze využít technologii dSpace, a to jak klasické PC s přídavným HW, tak specializovaná real-time HW řešení. Tato technologie se používá k řízení a simulace v reálném čase, kombinuje se výkonný hardware a automatickým generováním kódu ze Simulink modelu. dSpace je velmi dobrý nástroj pro rychlý vývoj (rapid-prototyping) řídících systémů a simulací hardware-in-the-loop (HIL).




**Obrázek 1.6:** Posloupnost kroků k řízení reálného systému dle technologie dSpace.



# KAPITOLA 2



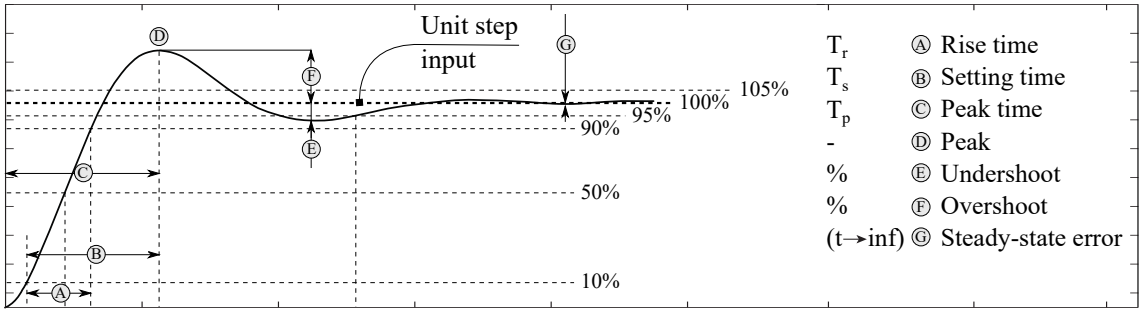
# APLIKACE VÝSLEDKŮ NA TESTOVACÍCH SOUSTAVÁCH

Kapitola aplikace výsledků na testovacích soustavách je zhodnocení výsledků použitých algoritmů v rámci teorii řízení nelineárního řízení komplexních systému a aplikování evolučních algoritmů na různé typy modelů a prezentuje výsledek této práce na dané téma. Celá kapitola je založena na autorových předešlých pracích (Seznam vlastních publikací - viz str. 32) a rozšiřuje způsob řešení o nové přístupy. Využívaní evolučních algoritmů v teorii řízení a dalších oborech technické praxe je stále relativně nový vědní obor, který poskytuje mnohé možnosti, jak zlepšit stávající přístupy k řešení optimalizačních úloh. Hlavní cil této sekce je vybrat takové modely a takové příklady na kterých bude jednoznačně demonstrováno jejich přednosti a jejich možnosti, jak při využívaní ve standartních přístupech, tak při aplikování nových přístupů řešení.

Nové přístupy v teorii řízení v této kapitole jsou zaměřeny k využití vícekriteriální optimalizace, kdy standartní postupy jsou zaměřeny většinou na jednu dominantní vlastnost a podle ní se provádí samotná optimalizace. U vícekriteriálního postupu se nezaměřuje jen na jednu vlastnost, ale snaží se příklad řešit ve více rovinách ohodnocení. Tento přístup přináší lepší výsledky a robustnější nastavení výsledných regulátorů. S využitím znalostí z řízení systémů je vícekriteriální optimalizace dále úspěšně aplikována i na řešení stability a navrhovaní struktury. Některé další postupy v této problematice jsou prezentovány v autorových předešlých pracích. Zhodnocení všech výsledků je na konci každé kapitoly a celkové porovnaní na přič všemi regulátory je v kapitole (sekce: 6 - viz str. 17) a představuje souhrnný výsledek této práce. Všechny modely a analýza jsou provedeny v aplikaci Matlab nebo Simulink, jak bylo prezentováno v předešlé praktické sekci.

## 2.1 Ohodnocení výsledků

K hodnocení kvality regulace je využita standardní metoda, a to metoda skokové odezvy uzavřeného regulačního obvodu. Tato metoda je zaměřena na zohlednění sedmi základních charakteristik, které jsou popsány na následujícím obrázku (obr: 2.1). K výpočtu hodnot jednotlivých charakteristik je použita funkce *stepinfo* zahrnuta v programu Matlab. Funkce je definována se základním nastavením pro všechny typy příkladů i pro všechny typy modelů kvůli jednoznačnosti porovnaní výsledků.




**Obrázek 2.1:** Ohodnocení přechodové charakteristiky.

$$fitness = A \times log_{10}(f_{ITAE}) + B \times log_{10}(f_{Overshot} + 1) + C \times log_{10}(f_{Wave} + 1) \quad (2.1)$$

## 2.2 Fitness funkce

Pro účely optimalizace byla vytvořena speciální fitness funkce s možností kompenzace různých obecných faktorů, které se obecně vyskytují při hodnocení přechodové funkce. Cela hodnotící funkce je zaměřena na přechodovou charakteristiku dle zadaného vstupu a skládá se primárně ze tří částí, dle obrázku (obr: 2.2).

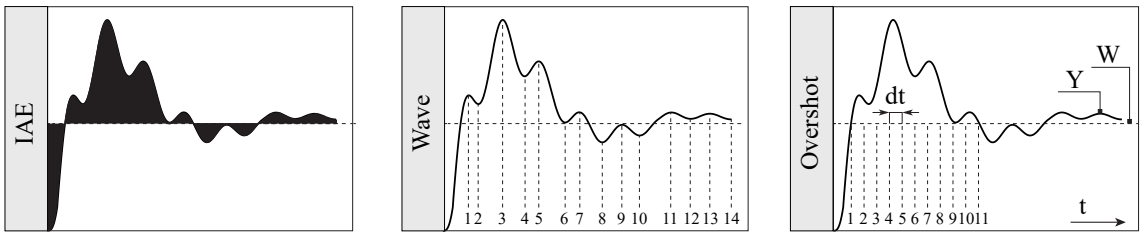
- **ITAE** - funkce itae je klasická hodnoticí funkce založena na funkci (IAE) v časové oblasti.
- **Wave** - kompenzační část hodnotící funkce k eliminaci kmitání přechodové funkce. Funkce je založena na sumarizaci kmitání a princip je znázorněný na obrázku (obr: 2.2) sekce $Wave$ spolu s definicí $f_{Wave}$ (vz: 2.2).
- **Overshot** - funkce overshot je kompenzátor přechodové funkce na překmit a je založena na časové sumarizaci hodnoty ve stavu kdy přechodová funkce je v překmitu. Princip je znázorněný na obrázku (obr: 2.2) sekce $Overshot$ a definice $f_{Overshot}$ (vz: 2.2).

$$\begin{aligned} f_{ITAE} &= \int_0^t t|e(t)|dt \\ f_{Wave} &= n \text{ kde } \frac{d}{dt}e(\{t_1 \dots t_n\}) = 0 \\ f_{Overshot} &= \sum_{i=1}^{n-1} \frac{(t_{i+1} - t_i)}{dt}, i+1 \text{ kde } e(\{t_1 \dots t_n\}) = W \end{aligned} \quad (2.2)$$

Kompletní fitness funkce je zapsaná ve tvaru (vz: 2.1) a jednotlivé charakteristické prvky jsou hodnotami logaritmické funkce při základu deset, kvůli eliminaci

nežádoucích hodnot. Jednotlivé prvky fitness funkce se dají ovlivňovat pomoci váhových proměnných ($A$, $B$, $C$), které určí váhu určité oblasti ve vícekriteriální oblasti optimalizace, rozmezí váhových proměnných je mezi $< 0, 1 >$.

Do výsledné fitness funkce se dají zanést i další podpůrné parametry specifické pro daný typ optimalizace a modelu jako je například filtrovaní přechodové funkce na extrémní hodnoty, nebo zadaný časový údaj náběhu, ale kvůli obecnosti a použitelnosti a porovnatelnosti pro všechny typy příkladů a modelů je využita verze fitness funkce dle (vz: 2.1).




**Obrázek 2.2:** Prvky fitness funkce.



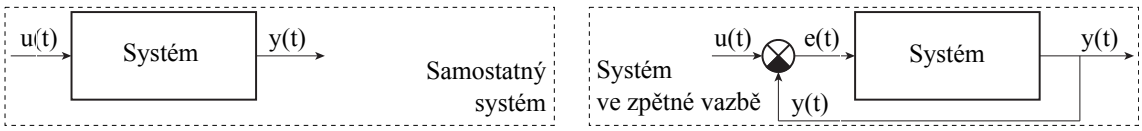
## 2.3 Popis testovacích modelů

Tato sekce je zaměřena na popis jednotlivých modelů v rámci tohoto testování. Popis je míněný jako stručný úvod k jednotlivým modelům a představení jednotlivých vlastností a typu využití modelů. U každého popisu, jestli je to možné, je uvedeno i referenční nastavení regulátoru typu (PID). Následující modely byly vybrány, aby co nejvíce obsahovaly největší množství vlastností při regulaci systémů, lineární systémy, nelineární modely reálných systémů až po diskrétní řízení. Pro jednoduchost jsou modely označovány jejich kódovými názvy a to:

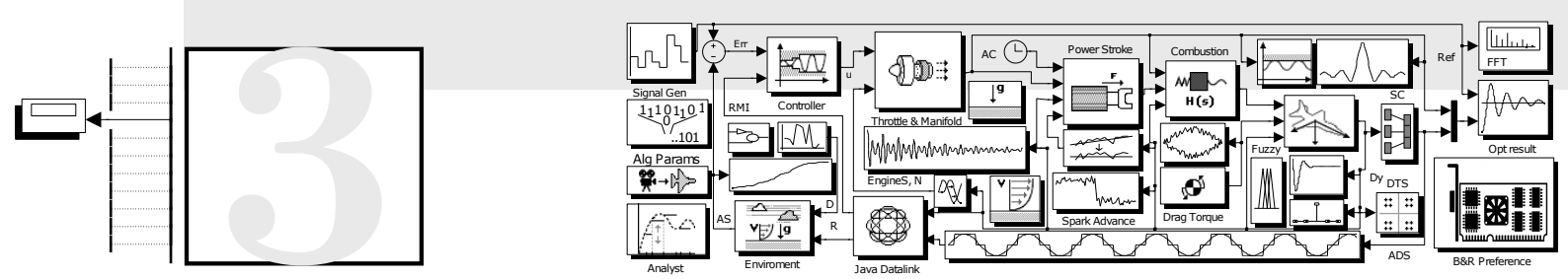
- Model čtvrtého řádu jako **system4**.
- Model čtvrtého řádu s nelineárním prvkem jako **system4delay**.
- Model kuličky v obruči jako **ballandloop**.
- Model magnetické levitace jako **maglev**.
- Logistická mapa jako **logisticmap**.
- Duffingova rovnice jako **duffing**.
- Yagi-Uda anténa jako **yagi**.

Pro modelování systému je využit software Matlab a pro určité typy optimalizace je využit jeho nástavba Simulink, dle nastavení popsaných u jednotlivých modelů, kvůli jednoznačnosti simulace.

Z důvodu určení stability systému je model testován na skokový signál pomocí obrázku (obr: 2.3), v samostatném obvodu nebo v obvodu ve zpětné vazbě.




**Obrázek 2.3:** Schéma zapojení systému k testovacím účelům na vstupní signál.




# KAPITOLA 3

# NASTAVENÍ PID REGULÁTORU

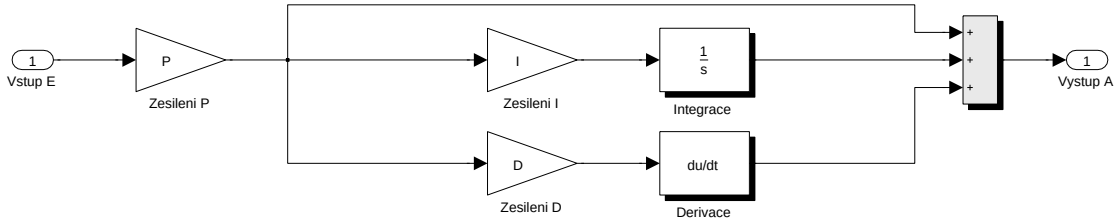


Prvním příkladem nastavení regulátoru je samostatné nastavení hodnot (PID). Tento typ regulátoru je zvolen jako nejpoužívanější typ s mozností odkázání na referenční nastavení, dle předchozí kapitoly. Tato kapitola je také míněna jako referenční bod k nastavení fitness funkce a jejích váhových hodnot (sekce: 2.2 - viz str. 10). Pro jednoduchost se berou čtyři základní nastavení, kdy váhové funkce zapínají jednotlivé prvky fitness funkce, tedy hodnota $A$, $B$, $C$ je buď 0 nebo 1 ($true/false$) a poslední možností, kdy jsou aktivovány všechny kombinace fitness funkce. Tedy kombinace nastavení funkce jsou tato, dle kódového označení F100 až F111.

V rámci testování jsou využité všechny kombinace jen pro první model, tedy model $system4$, k porovnání vlivu jednotlivých váhových proměnných. Další modely už využívají jen nejvíce komplexní nastavení fitness funkce $F111$ k porovnaní s referenčním nastavením.

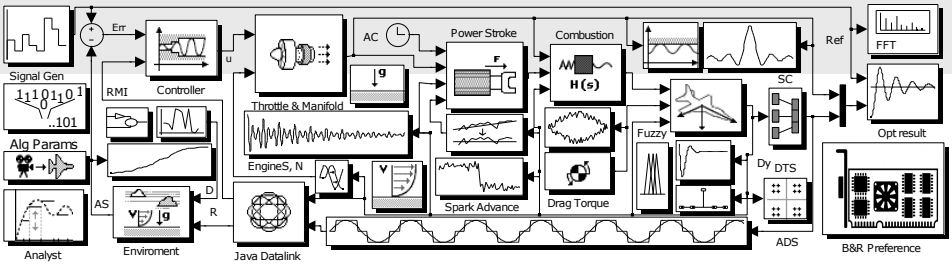
V poslední řadě tato sekce slouží jako porovnaní vhodnosti jednotlivých optimalizačních algoritmů pomocí statistiky na 100 opakovaní pro každý chod optimalizace.

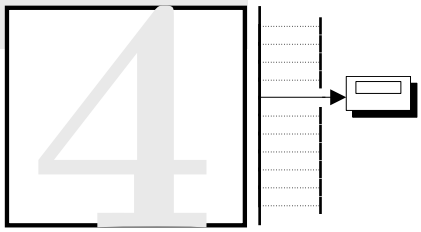
Všechny modely budou řízeny pomocí regulátoru (PID) v časové rovině pomocí vzorce (vz: 3.1), kromě prvního modelu *system4*, který využívá mírně upravenou strukturu (vz: 3.1) v integračním zesilovači viz model (model: 3.1).




**Model 3.1:** Struktura upraveného PID regulátoru.

$$u(t) = r_0 \left( e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{d}{dt} e(t) \right) \tag{3.1}$$



# KAPITOLA 4

# NASTAVENÍ FUZZY PID REGULÁTORU

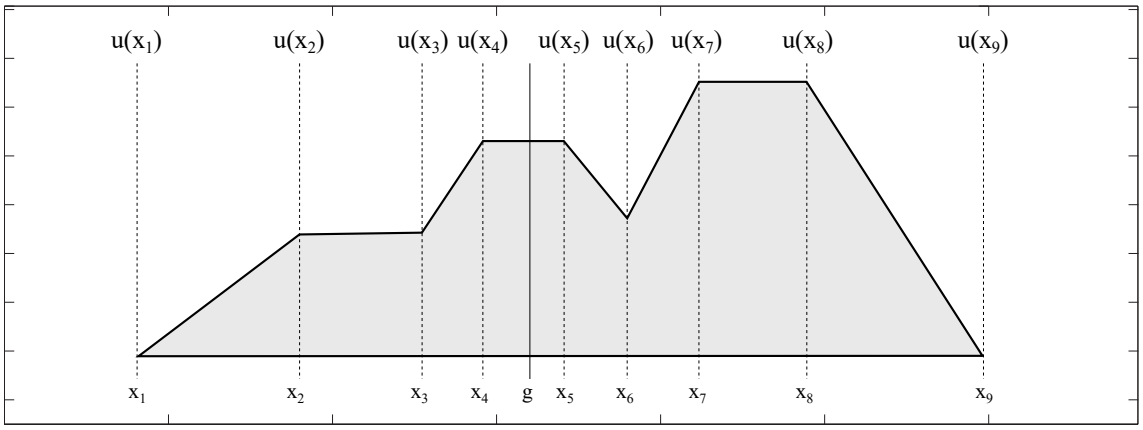
Druhý příklad nastavení regulátoru je nastavení fuzzy logického kontroléru s využitím znalostí z (PID) regulátoru. Fuzzy kontrolér spadá do oblasti pokročilého nastavení a oproti samostatnému (PID) má schopnost být velice robustním regulátorem. Tato kapitola kombinuje znalosti z předchozí kapitoly, kdy struktura optimalizovaného regulátoru je kombinací PID struktury (kde se využívají jak derivační a integrační člen) a fuzzy logiky. Tento přístup dává vetší prostor nastavení regulátoru oproti klasickému PID, kdy můžeme vhodně kombinovat funkce příslušnosti, z hlediska jejich tvarů a pozice a vstupních/výstupních zesilovačů k fuzzy kontroléru. Celková struktura optimalizovaného regulátoru je zobrazena na modelu (model: 4.1) a je využívána pro všechny typy modelů v této kapitole.

Struktura samotného fuzzy kontroléru obsahuje tři vstupy, každý s třemi funkcemi příslušnosti, popsanými Gaussovou rovnicí přímky, pozicí a rozptylem. Jeden výstup s jedenácti funkcemi příslušnosti, tak abychom popsali všechny možnosti vstupů a výstupů. Všechny funkce příslušnosti mají rozsah od (-100 do 100). Celková struktura fuzzy regulátoru je tedy popsaná dvaceti-sedmi bázovými pravidly. Celková komplexnost regulátoru tedy dává možnost nastavení až čtyřiceti parametrů (dva pro každou funkci příslušnosti). K vypočítání hodnoty výstupu z báze pravidel se využívá metody *centroid*. Centriod je funkce defuzzifikace, kde se výstup ztotožňuje s těžištěm celkové plochy všech pravidel (vz: 4.1) a (obr: 4.1). Funkční pravidla jsou popsána logickou operací $AND$, tedy minimální funkcí z výsledku funkčního pravidla, operátor $AND$ je popsán ve vzorci (vz: 4.2). Fuzzy regulátor je typu Mandani.

Z hlediska optimalizace je celý proces nastavení rozdělen do tří způsobů výsledných postupů, na kterých je znázorněn vliv jednotlivých nastavení. První způsob nastavení je čistý fuzzy PID, kdy se nastavují jen vstupní a výstupní zesílení, při lineárním rozložení funkcí příslušnosti. První způsob by měl ukázat kvalitativně podobné výsledky jak předchozí čisté nastavení (PID) regulátoru. Druhý způsob je

nastavení funkcí příslušnosti s konstantním vstupním a výstupním zesílením na hodnotu 1, Nastavení probíhá na Gaussových funkcích, kdy se nastavuje jejich rozptyl a jejich pozice v jednotlivých vstupech nebo výstupu. Poslední možností nastavení fuzzy regulátoru je kompletní nastavení, jak vstupních zesilovačů, tak funkcí příslušnosti v jedné optimalizaci. Tento způsob je nejvíce komplexní z hlediska nastavení regulátoru a má mnohem vetší stavový prostor řešení než předcházející PID regulátor.

$$g = \frac{\sum_{i=1}^{n} x_i u(x_i)}{\sum_{i=1}^{n} u(x_i)} \tag{4.1}$$




**Obrázek 4.1:** Výpočet defuzzyfikace pomoci funkce centroid.

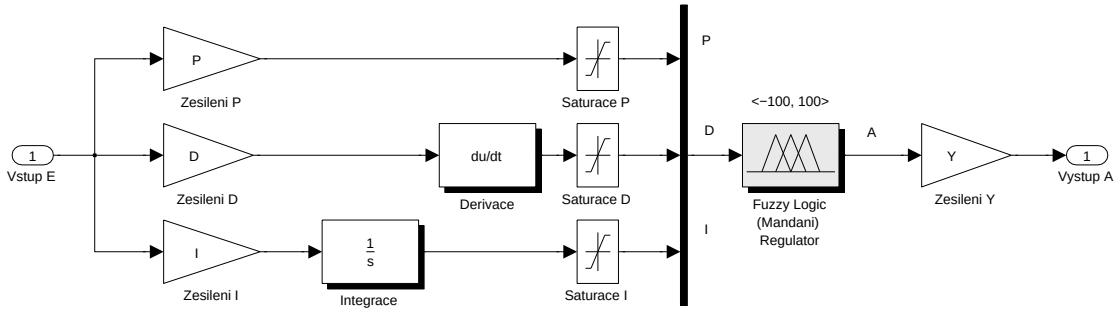


Stejně jak přechozí část i zde je využitý kompletní způsob nastavení jen u prvního modelu (Model čtvrtého řádu), který představuje referenční bod k ostatním nastavením, kdy se používají dále jen komplexní FPID-FPnastavení fuzzy PID.

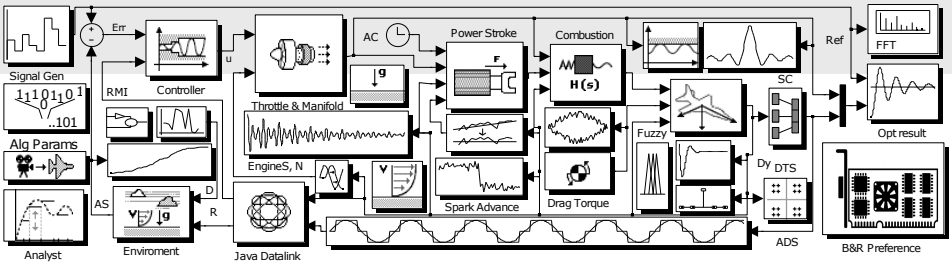
$$\mu_C(x) = \mu_{A \wedge B}(x) = min\left(\mu_A(x), \mu_B(x)\right) \tag{4.2}$$

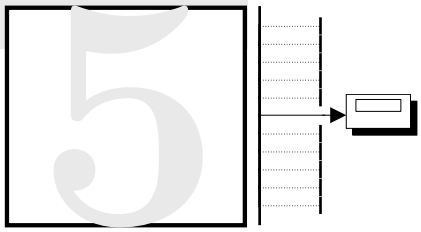
Fuzzy regulátor je velice komplexní regulátor, který nabízí množství druhů nastavení oproti klasickému (PID). Další mozností je nastavení počtu funkcí příslušnosti s kombinací s vice druhů funkcí nebo generovaní báze pravidel dle genetických algoritmů spolu s optimalizací váhových proměnných pro jednotlivá pravidla.

Obecně fuzzy nastavení je zejména vhodné pro hledání robustního kontroléru, v této oblasti fuzzy regulace vykazuje velice dobré výsledky.




**Model 4.1:** Struktura FUZZY PID regulátoru.



# KAPITOLA 5

# GENEROVANÍ STRUKTURY REGULÁTORU

Poslední příklad v regulaci systémů je regulace dle vytvořených regulačních zákonů pomocí gramatické evoluce s postupným doladěním v (HC12) algoritmu. Generovaní regulačních zákonů dává největší svobodu při navrhovaní výsledného regulátoru a závisí na kvalitě používané gramatiky. Z hlediska tohoto přikladu se používá Backus-Naurova forma gramatiky ve dvou základních verzích, první verze je pro systémy (model čtvrtého řádu, model čtvrtého řádu s nelineárním prvkem, model kuličky v obruči) a je založena na generovaní pravidel v Matlab syntaxi, druhá verze je pro (model magnetické levitace) a je založena na generovaní simulinkovských schémat.



Dvě verze gramatik se pozívají z důsledku prezentace, kdy oba dva druhy je možné úspěšně použít k řešení problému a z důsledku nutnosti v modelu *maglev* použít regulátor v simulinkovské syntaxi. Kvalitativně oba dva způsoby jsou na tom podobně, kdy oba způsoby dokáži najit optimalizovaný regulátor, jen zápis v Matlab syntaxi je rychlejší na vypočet regulace, a proto je vhodnější pro běžné použití.

Matlab verze gramatického zápisu je znázorněna na rovnici (gramatika: 5.2) a hlavním prvkem gramatiky je tvorba obecných přechodových funkcí pomocí zápisu rovnice (vz: 5.1). Kvůli možnosti generovaní až dokonalých regulátorů je nutno zavést omezující podmínky pro výsledný regulátor. Hlavní podmínkou je vygenerovaní fyzikálně realizovatelného systému, tedy výsledný systém bude obsahovat více pólů než nul ($m \leq n$) v rovnici přenosu. Další podmínkou je vygenerovaní reálného regulátoru z omezení maximálního výstupních i vstupních koeficientů diferenciální rovnice na hodnotu 100.

Gramatika popsaná bloky v Simulink syntaxi je založená na generovaní klasického regulátoru s integrační a derivační složkou spolu s obecnými matematickými operacemi. Omezení pro tento typ regulace je stejný jak pro verzi v Matlab zápisu. Celková gramatika je zobrazena pomocí rovnic (gramatika: 5.1).

Poslední části této kapitoly je ladění výsledných regulačních zákonů dle algoritmu

```
       N = { init, node, block_input_1, block_input_2, inputs, consts }
       T = { +, -, *, /, input_1, integrator, derivator, 1 - 9 } | S = { init }
<init>          ::(<node><block_input_1>) | (<node><node><block_input_2>)
<node>          ::(<init>) | (<inputs>) | (<consts>)
<block_input_1> ::integrator | derivator
<block_input_2> ::+ | - | / | *
<inputs>        ::input_1
<consts>        ::1 | ...| 9
```

**Gramatika 5.1:** Obecná gramatika k regulaci systémů v prostředí Simulink.

```
         N = { expr, expr2, op, tr_func, vector, var }
         T = { +, -, *, /, X, -1, 0, 1 - 9 } | S = { expr }
<expr>   ::(<tr_func><op><tr_func>) | (<var><op><tr_func>)
<expr2>  ::(<expr2> <op> <expr2>) | <var> | (<var> <op> <var>) | <vector>
<op>     ::+ | - | / | *
<tr_func>::tf(<vector>, <vector>) | tf(<vector>, <vector>)<op><expr>
<vector> ::[<expr2>, <expr2>]
<var>    ::-1 | 0 | 1 | ...| 9
```

**Gramatika 5.2:** Obecná gramatika k regulaci systémů v prostředí Matlab.

(HC12). Ladění je prováděno v důsledku, kdy gramatická evoluce dokáže najit použitelný regulační zákon, ale jemné doladění všech koeficientů je vhodné využít odlišný algoritmus k numerické optimalizaci. Pro Matlab verzi gramatiky se dolaďují výsledné póly a nuly přechodové rovnice dle zadaného rozsahu a pro Simulink verzi je ladění prováděno na vytvořených zesilovačích vstupního signálu. Každý model této kapitoly má zobrazený, jak výraz po gramatické evoluci bez uprav, tak výraz, který je použít v algoritmu (HC12) na finální optimalizaci.

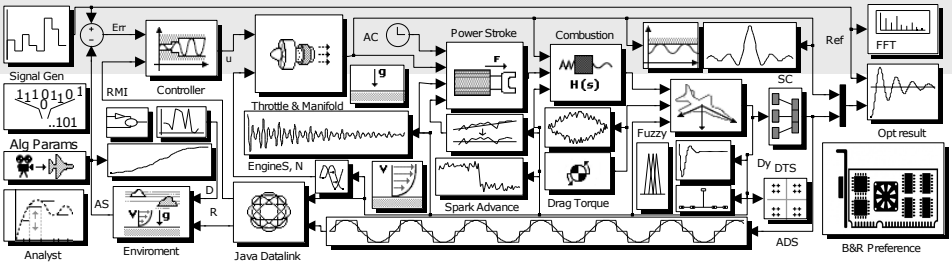


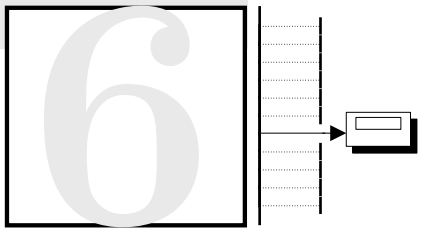
Oproti předešlým kapitolám se v této kapitole používají všechny způsoby regulace pro všechny modely, je to hlavně kvůli rozmanitosti generovaných struktur a výsledné dolaďovací optimalizaci. Kvůli zprůhlednění výsledků a zachovaní jmenné konvenci se pozívají kódové označeni jednotlivých způsobů a jsou to tyto:

- Generovaní regulačních zákonů pomocí gramatické evoluce jako **Version**.
- Doladění finálních koeficientů vygenerovaného regulačního zákona jako **Tuning**.

Generovaní vlastních regulačních pravidel obecně dokáže najít velmi kvalitní systém k regulaci, který v mnohém předčí standardní regulátory jako je třeba (PID). Tento systém má, ale nevýhodu vysoké výpočetní náročnosti a je tedy vhodný jen v podmínkách kdy chceme najit specifický regulátor s přesně definovanými podmínkami. Takové podmínky můžou být, jako v této kapitole, fyzikální realizovatelnost, maximální koeficienty rovnice anebo třeba jen databanka možných součástek, které mužem použít při realizovaní regulátoru. Tento typ nastavení tedy dává velice velké moznosti při nastavení výsledné regulace. Hlavně ve spojitosti s doladěním výsledné struktury dle optimalizačního algoritmu (HC12), s tímto finálním procesem se zlepši kvalita regulátoru na úroveň velmi robustního řízení.

$$G(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \dots b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} \dots a_1 s + a_0} \ (m \leq n) \tag{5.1}$$



# KAPITOLA 6

# POROVNÁNÍ VÝSLEDKŮ PRO PŘEDCHOZÍ MODELY

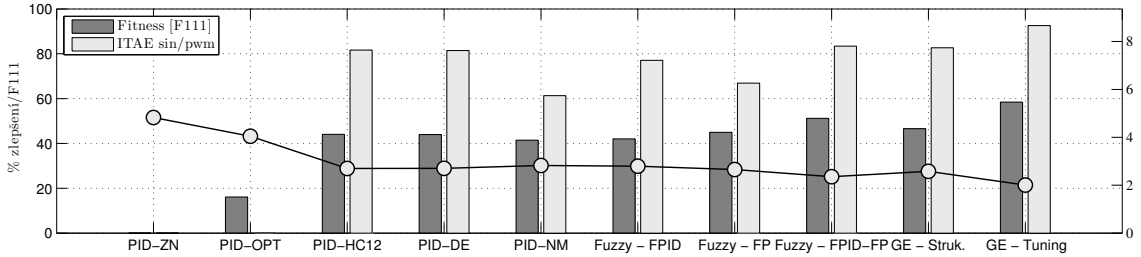
Hodnocení konečných výsledků jsou porovnány na základě fitness funkce ze sekce (sekce: 2.2 - viz str. 10) v podobě $A = 1$, $B = 1$ a $C = 1$ v označení *F111* ke standartnímu nastavení pro jednotlivé systémy, které je prezentované v popisu jednotlivých systémů ze sekce (sekce: 2.3 - viz str. 11).

Dále je porovnáván dynamické chování systému na základě (ITAE) funkce (vz: 2.2) k základnímu nastavení SIN/PWM přechodové funkci. Fitness *F111* je prezentované graficky pro všechny opakování algoritmu v daném typu regulace (100 opakování) jako hlavní kritérium hodnocení v této práci. Výsledky jsou rozdělené do třech základních částí, první představuje regulaci dle (PID) regulátoru pro algoritmy HC12, diferenciální evoluci a Nelder-Mead metodu ze sekce (sekce: 3 - viz str. 12). Druhá část výsledů je fuzzy regulace pro typy *FPID*, *FP* a *FPID-FP*, pro první systém (model čtvrtého řádu), ostatní systémy se prezentují ve formě *FPID-FP*, pro algoritmus (HC12), výsledky ze sekce (sekce: 4 - viz str. 13). Poslední typ výsledů je generování struktury samotného regulátoru, sekce (sekce: 5 - viz str. 15), a představují výsledky z gramatické evoluce a následující doladění v podobě algoritmu HC12 na vybraných parametrech struktury regulátoru z (GE) výpočtu. Finální hodnoty výsledů jsou dále zobrazeny v tabulce pro nejlepší hodnocení, pro medián a pro standardní odchylku rozptylu ze všech opakování algoritmu, jak pro *F111* tak pro dynamické chování SIN/PWM funkce.

Poslední graf pro jednotlivé porovnání představuje průběh nejlepší fitness hodnoty *F111* pro jednotlivé typy regulace a zobrazuje procentuální zlepšení těchto typů regulátorů / optimalizace oproti základnímu nastavení. Procentuální hodnoty jsou zobrazeny jak pro *F111* fitness funkci tak pro ohodnocení SIN/PWM přechodové funkce ve formě základní ITAE funkce. Následující výsledky ukazují vhodnost použití evolučních algoritmů na hledání optimálních regulátorů pro různé typy systémů. První část prezentuje optimalizaci (PID), kdy při použití algoritmů (HC12), (DE) a (NM) dokáží nalézt vhodnější řešení k dané fitness funkci oproti standart-

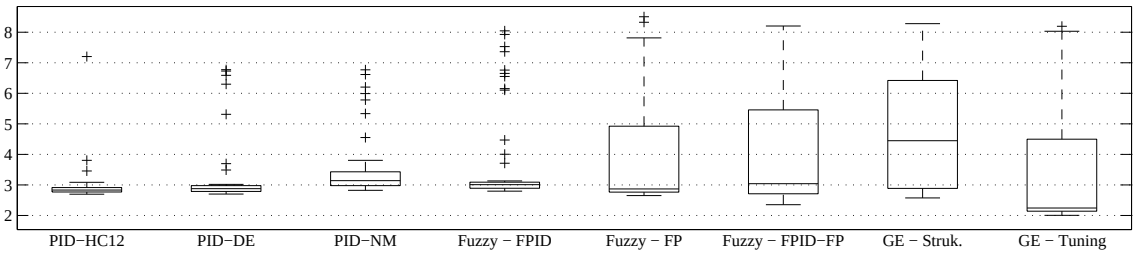
ním metodám nastavení. Další výhodou je vhodnost použití tohoto postupu při řešení nelineárních systémů, kdy není možno použít standartní metody typu (ZN) a další. Při výpočtu fuzzy regulace jsou prezentované obdobné výsledky jak pro (PID) s lepší dynamickou odezvou při SIN/PWM přechodové funkci. Posední část je vytváření vlastních struktur regulátoru a prezentuje vhodnost metody v případě použití atypického regulátoru, který nemá pevně danou strukturu nastavení. Při správném navolení definice gramatiky generování struktury dosahují konečné výsledky obdobné výsledky jak nejlepší nastavení (PID) regulátoru. Poslední možnost je následné doladění výsledné struktury (GE) regulace pomocí (HC12) algoritmu na vybraných parametrech. Kdy tato metoda představuje nejlepší výsledky jak pro fitness *F111* tak z hlediska dynamické přechodové funkce.

Výsledné hodnoty, pro všechny modely, vytvořených regulátorů na základě jejich fitness hodnoty jsou prezentovány na následujících grafech a statistických porovnáních. Model čtvrtého řádu: reprezentují grafy (graf: 6.1), (graf: 6.2), (graf: 6.3), (graf: 6.4); model čtvrtého řádu s nelineárním prvkem: (graf: 6.5), (graf: 6.6), (graf: 6.7); model kuličky v obruči: (graf: 6.8), (graf: 6.9), (graf: 6.10); model magnetické levitace: (graf: 6.11), (graf: 6.12), (graf: 6.13). Z porovnaní vyplivá nejlepší regulátor z hlediska procentuálního zlepšení oprati standardnímu nastavení a to optimalizace na základě (GE) algoritmu a následnému doladění pomocí (HC12). Jednotlivé hodnoty jsou závislé na typu hodnoticí funkce a multikriteriální optimalizace.

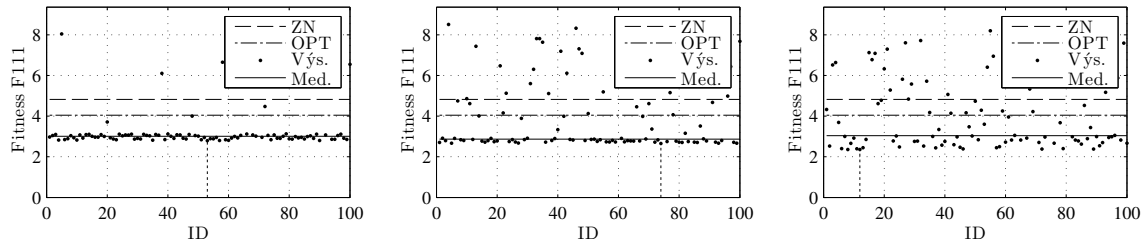



**Graf 6.1:** Hodnoty fitness funkce pro všechny typy regulace a procentuální zlepšení oproti základnímu nastavení regulátoru [model čtvrtého řádu].

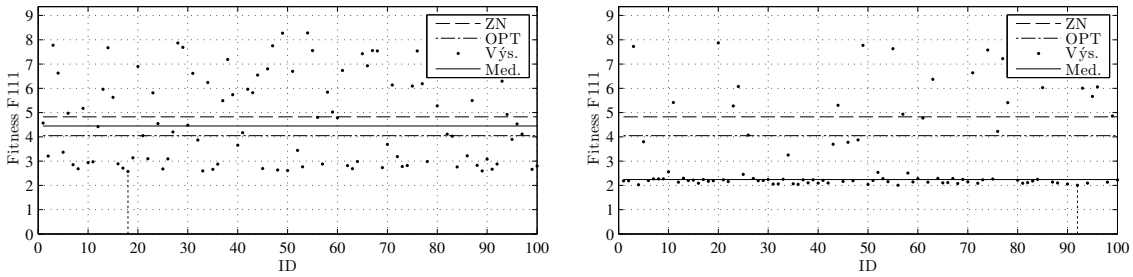




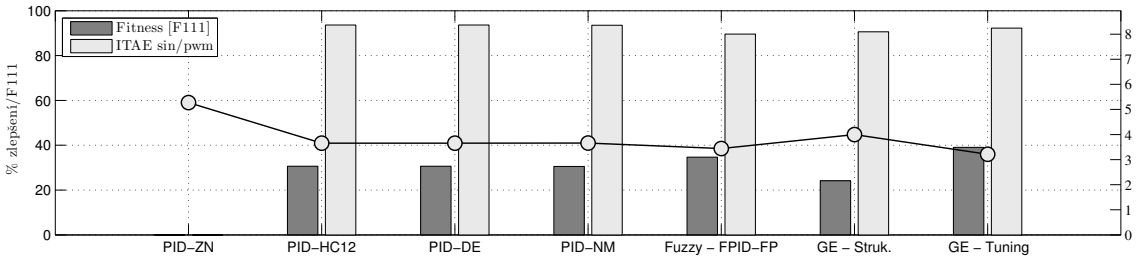

**Graf 6.2:** Box graf hodnot fitness funkce pro všechny typy regulace [model čtvrtého řádu].



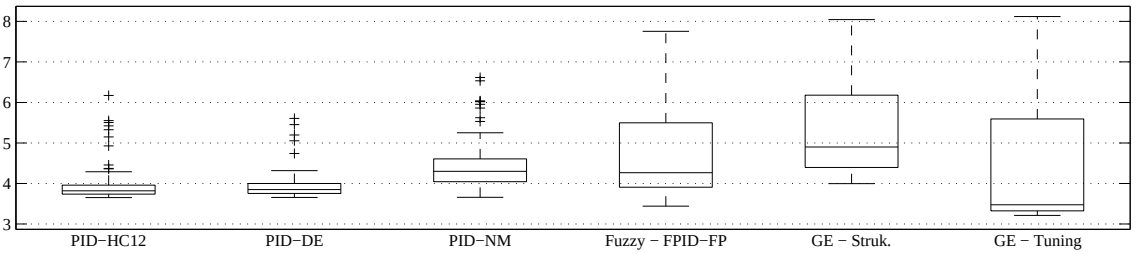

**Graf 6.3:** Hodnoty všech fitness hodnocení [F111] k základnímu nastavení pro výpočet [FUZZY] regulátoru. Pořadí grafů: (a: Fuzzy - FPID), (b: Fuzzy - FP) a (c: Fuzzy - FPID-FP).



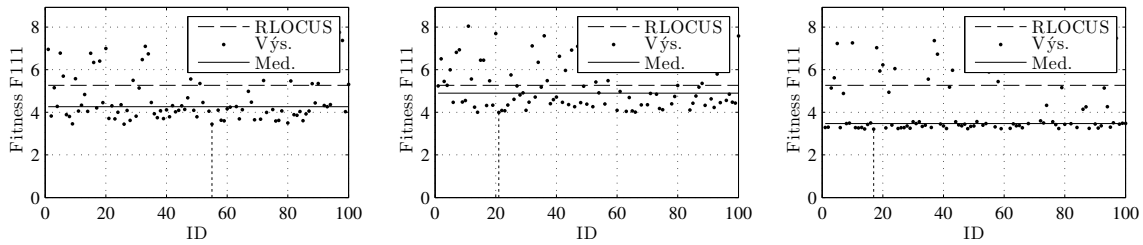

**Graf 6.4:** Hodnoty všech fitness hodnocení [F111] k základnímu nastavení pro výpočet [GE-Tuning] regulátoru. Pořadí grafů: (a: GE - Struk.) a (b: GE - Tuning).



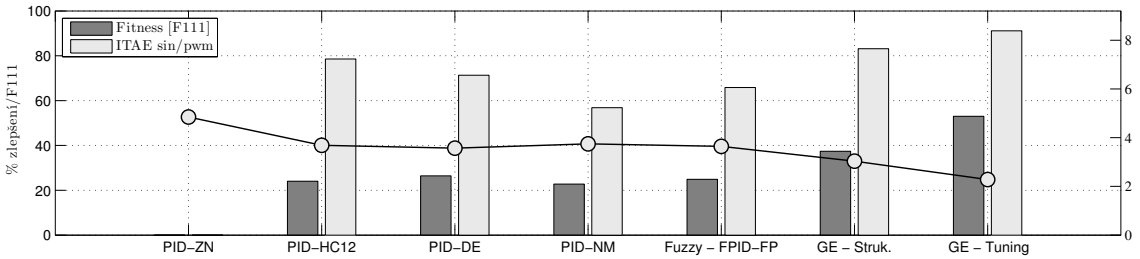

**Graf 6.5:** Hodnoty fitness funkce pro všechny typy regulace a procentuální zlepšení oproti základnímu nastavení regulátoru [model čtvrtého řádu s nelineárním prvkem].



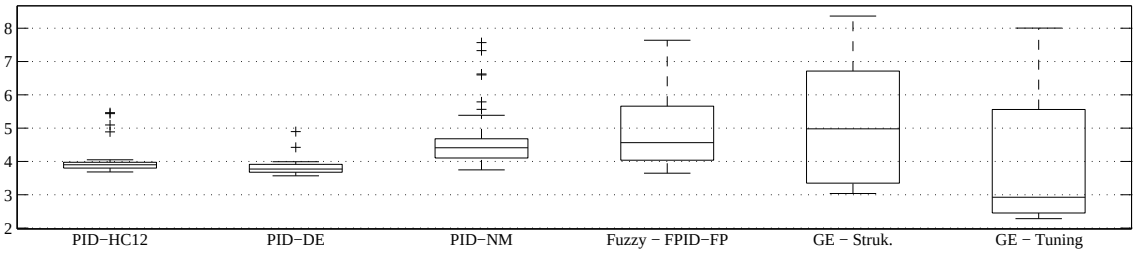

**Graf 6.6:** Box graf hodnot fitness funkce pro všechny typy regulace [model čtvrtého řádu s nelineárním prvkem].



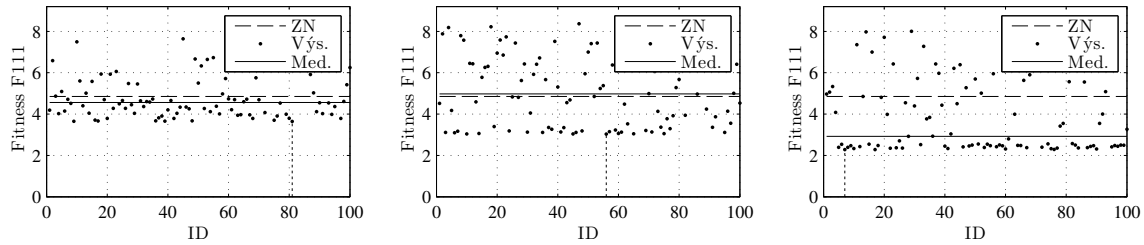

**Graf 6.7:** Hodnoty všech fitness hodnocení [F111] k základnímu nastavení pro výpočet [FUZZY-GE-Tuning] regulátoru. Pořadí grafů: (a: Fuzzy - FPID-FP), (b: GE - Struk.) a (c: GE - Tuning).



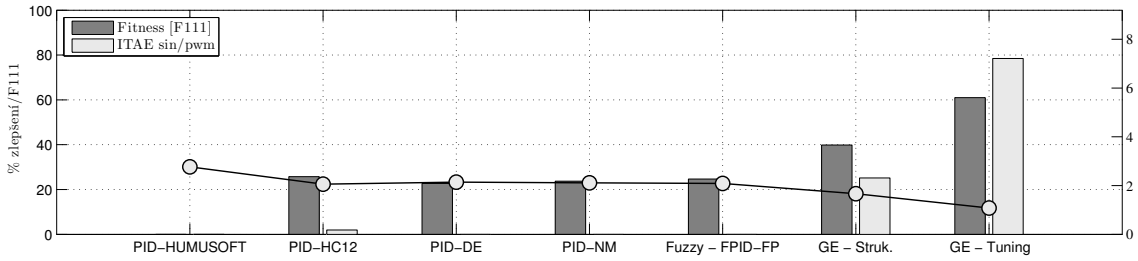

**Graf 6.8:** Hodnoty fitness funkce pro všechny typy regulace a procentuální zlepšení oproti základnímu nastavení regulátoru [model kuličky v obruči].



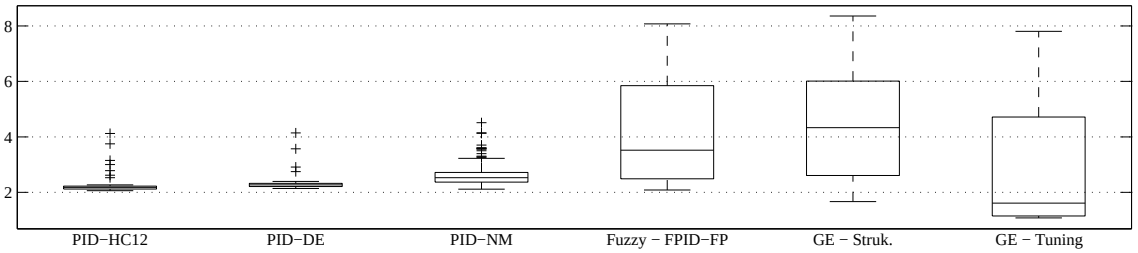

**Graf 6.9:** Box graf hodnot fitness funkce pro všechny typy regulace [model kuličky v obruči].



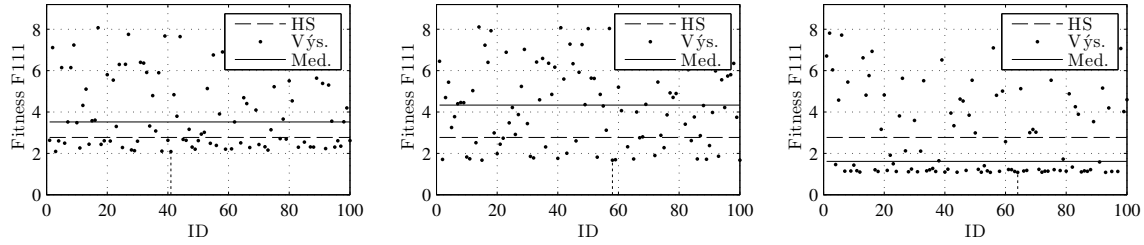

**Graf 6.10:** Hodnoty všech fitness hodnocení [F111] k základnímu nastavení pro výpočet [FUZZY-GE-Tuning] regulátoru. Pořadí grafů: (a: Fuzzy - FPID-FP), (b: GE - Struk.) a (c: GE - Tuning).




**Graf 6.11:** Hodnoty fitness funkce pro všechny typy regulace a procentuální zlepšení oproti základnímu nastavení regulátoru [model magnetické levitace].

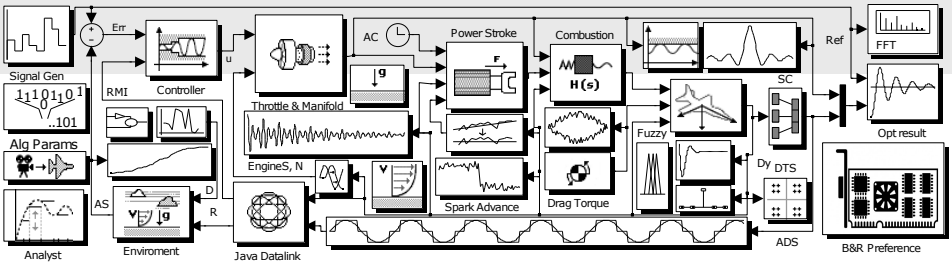


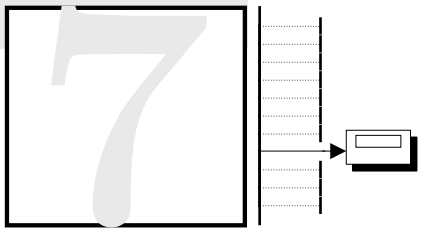

**Graf 6.12:** Box graf hodnot fitness funkce pro všechny typy regulace [model magnetické levitace].




**Graf 6.13:** Hodnoty všech fitness hodnocení [F111] k základnímu nastavení pro výpočet [FUZZY-GE-Tuning] regulátoru. Pořadí grafů: (a: Fuzzy - FPID-FP), (b: GE - Struk.) a (c: GE - Tuning).

Kompletní výsledky jsou prezentované v plné verzi této disertační práce společně s detailním popisem nastavení všech algoritmů a jednotlivých typů regulace. Plná verze práce dále obsahuje i stručný popis regulace nelineárních systémů a multikriteriální optimalizace.




# KAPITOLA 7

# PŘÍKLAD STABILIZACE LOGISTICKÉ MAPY

Sekce stabilizace chaosu se zabývá vytvořením stabilizačních pravidel pro chaotický systém, tedy na logistickou mapu v kontrolním bode $r = 3.8$. Systémy v chaotickém stavu jsou stabilizovány pomoci malých akčních zásahů na aktuálním stavu sytému v iteraci k dosažní požadované hodnoty. Požadovaná hodna je určena dle fixních bodů (UPO), výpočet těchto bodů je znázorněn v grafech (graf: 7.1a) - (graf: 7.1c).

Obecný využívaný způsob stabilizace logistické mapy je metoda zpožděné zpětné vazby (Pyragasova metoda). Pro názorný příklad stabilizace je tedy nastavený akční zásah pomocí Pyragasova metody jako základní regulátor k porovnání výsledků s rovnicemi vygenerovanými dle gramatické evoluce. Experiment by měl dokázat využití obecných pravidel dle (GE) pro nastavení základního stabilizačního pravidla jako vhodná alternativa k Pyragasově metodě i s pozdějším doladěním dle metod soft computing, zde metoda (HC12). Základní popis metody je ve vzorci (vz: 7.1), předpokládaný systém $P$ obsahuje proměnou hodnotu $x$ a hodnotu $F$, která značí externí řízený parametr (stabilizátor). Systém bez stabilizace obsahuje hodnotu $F$ rovna 0, tedy bez akčního zásahu.



$$\frac{dx}{dt} = P(x) + F(t) \tag{7.1}$$

Fitness hodnota k optimalizaci stabilizace je počítána pomocí základní funkce (IAE) v časové (iterační) oblasti (vz: 2.2) pro jeden stabilizující orbit. K vice orbitů je využita obdobně (ITAE) metoda, ale s úpravou pro vice požadovaných hodnot s atraktorem hodnoty.

Stabilizace chaosu v tomto případě je rozdělena na tři části. První část je stabilizace dle Pyragasové metody s využitím optimalizační metody (HC12), druhá část je vytvoření vlastních pravidel pomoci gramatické evoluce s využitím základních hodnot regulátoru (TDAS)/(ETDAS) a poslední část je doladění předchozí části dle algoritmu (HC12). Pro vytváření vlastních stabilizačních pravidel jsou navrženy

```
        N = { start, weight, expr, expr2, pre_op, op, op2, var, var2}
        T = { +, -, *, /, 1 - 9} | S = { start }

<start> ::(<weight>*<expr><op><expr>)
<weight>::(<weight><op><weight>) | (<var><op><weight>) | (<weight><op><var>)|
          (<var><op><var>)
<expr>  ::<expr> | (<expr><op><expr>) | (<var><op><expr>) | <pre_op><op><expr>|
          <pre_op><op><pre_op>
<pre_op>::x( index_check( n - <expr2> ) )
<expr2> ::(<var2> <op2> <var2>) | <var2>
<op>    ::+ | - | / | *
<op2>   ::+
<var>   ::1 | ...| 9
<var2>  ::1 | <var2>
```

**Gramatika 7.1:** Nastavení gramatiky k regulaci chaosu pro nízký počet bodů (UPO).

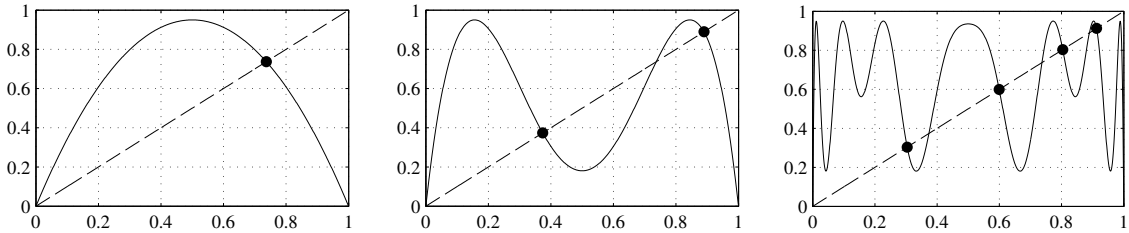
```
        N = { start, weight, expr, expr2, pre_op, op, op2, var, var2}
        T = { +, -, *, /, 1 - 9} | S = { start }

<start> ::<weight>*(<weight>*<expr><op><weight>*<expr>)
<weight>::(<weight><op><weight>) | (<var><op><weight>) | (<weight><op><var>)|
          (<var><op><var>)
<expr>  ::<expr> | (<expr><op><expr>) | (<var><op><expr>) | <pre_op><op><expr>|
          <pre_op><op><pre_op>
<pre_op>::x( index_check( n - <expr2> ) )
<expr2> ::(<var2> <op2> <var2>) | <var2>
<op>    ::+ | - | / | *
<op2>   ::+
<var>   ::1 | ...| 9
<var2>  ::1 | <var2>
```

**Gramatika 7.2:** Nastavení gramatiky k regulaci chaosu pro 4 body (UPO).

dvě základní verze gramatik. První verze je znázorněna na popisu (gramatika: 7.1) a hodí se k regulovaní jen nízkého počtu stabilizujících (UPO), zde je používána na jeden a dva fixní body, gramatika generuje pravidla, která pro další optimalizaci zahrnují celkem tři základní body a to $F_{max}$, $X_1$ a vytvořený $param_1$. Druha verze gramatiky je opravená první verze s přidáním vice optimalizujících parametrů na $param_1$, $param_2$ a $param_3$ a je popsaná dle (gramatika: 7.2). Sekce generovaní pravidel obsahuje jak formu po vygenerovaní pravidla, tak formu po úpravě k další optimalizaci dle (HC12) algoritmu.

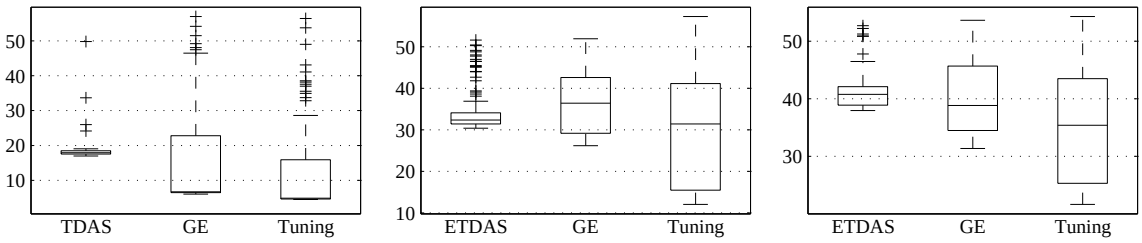





**Graf 7.1:** Grafický výpočet fixních bodů pro logistickou mapu v řádu 1, 2 a 4. Pořadí grafů: (a: LM 1. řádu v parametru r (3.8), $\lambda$ 45.), (b: LM 3. řádu v parametru r (3.8), $\lambda$ 45.) a (c: LM 4. řádu v parametru r (3.8), $\lambda$ 45.).

## 7.1 Porovnání jednotlivých výsledků

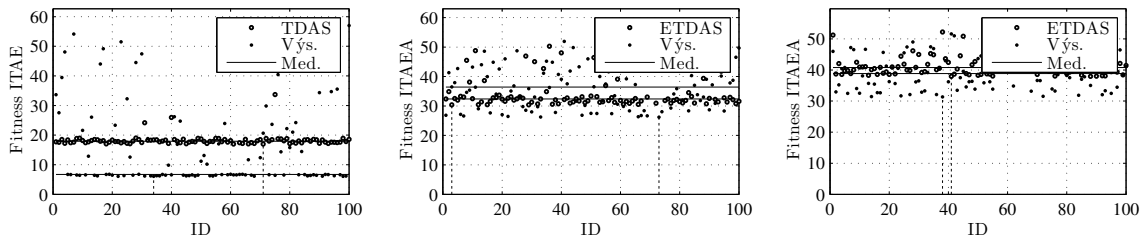
V této sekci porovnání výsledků je shrnuto využití jednotlivých algoritmů použitých v této práci na stabilizaci deterministického chaosu a strukturou vyhází z kapitoly (sekce: 6 - viz str. 17). Rozdělení jednotlivých výsledků a jejich grafická analýza pomocí box grafů je prezentována pro výsledky fitness funkce (graf: 7.2a), (graf: 7.2b) a (graf: 7.2c). Optimalizace stability deterministického chaosu je složitá procedura a závisí na samotné chaosové funkci a funkci použité k výsledné stabilizaci. Standartní požívaná funkce ke stabilizaci chaosu je považována Pyragasova metoda ve formátu (TDAS) pro jeden (UPO) fixní bod nebo (ETDAS) pro více (UPO) fixních bodů, proto je tato metoda požitá jako referenční nastavení dle (HC12) algoritmu. V této práci je hlavně rozvíjená možnost stabilizace chaosu dle gramatické evoluce se speciálně nastavenou gramatikou stabilizace (gramatika: 7.1) a (gramatika: 7.2). Grafické zobrazení výsledků pro všechny typy (UPO) stabilizace je dle grafů (graf: 7.3a), (graf: 7.3b) a (graf: 7.3c). Následující doladění dle metody (HC12) na vygenerovaných funkčních pravidel z předešlé části je reprezentované grafech (graf: 7.4a), (graf: 7.4b) a (graf: 7.4c).



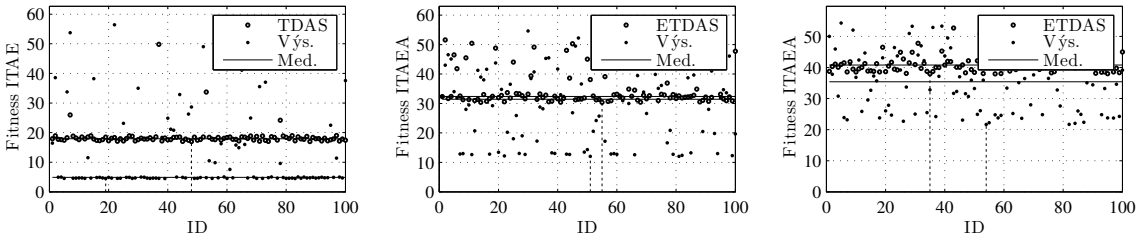

**Graf 7.2:** Box grafy pro hodnoty fitness. Pořadí grafů: (a: UPO1), (b: UPO2) a (c: UPO4).

Tento postup využití gramatické evoluce a následné požití doladovacích postupů dle (HC12) se velmi dobře uplatnil v předešlé části k regulaci nelineárních systémů a pro stabilizaci deterministického chaosu také vykazuje velice dobré výsledky, jak prezentuje graf (graf: 7.5), který zobrazuje jednotlivé procentuální zlepšení pro všechny (UPO) fixní body k základnímu nastavení (Pyragasova metoda dle algoritmu HC12) pro gramatickou evoluci a následné doladění dle (HC12) algoritmu. Graf také zobrazuje jednotlivé nejlepší ohodnocení pro fitness (vz: 2.2) nebo speciální ITAE funkce pro více fixních bodů.

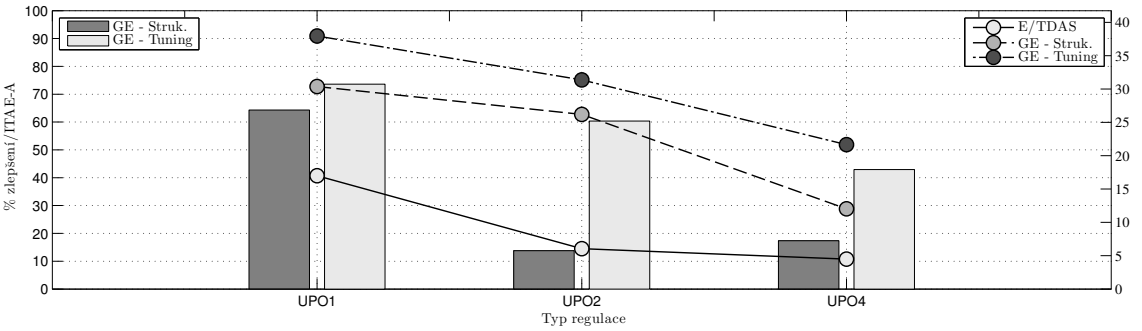
Výsledné hodnoty jsou v tabulce (tab: 7.1) pro nejlepší hodnoty fitness, medián a standartní odchylku rozptylu pro všechna opakování výpočtů.




**Graf 7.3:** Hodnoty všech fitness hodnocení [ITAE/ITAEA] k základnímu nastavení pro výpočet [GE] regulátoru. Pořadí grafů: (a: UPO1 - TDAS - GE), (b: UPO2 - ETDAS - GE) a (c: UPO4 - ETDAS - GE).




**Graf 7.4:** Hodnoty všech fitness hodnocení [ITAE/ITAEA] k základnímu nastavení pro výpočet [Tuning] regulátoru. Pořadí grafů: (a: UPO1 - TDAS - Tuning), (b: UPO2 - ETDAS - Tuning) a (c: UPO4 - ETDAS - Tuning).



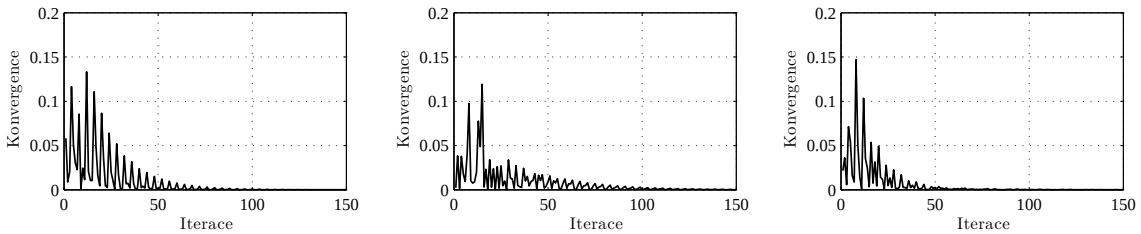

**Graf 7.5:** Hodnoty fitness funkce pro všechny typy regulace a procentuální zlepšení oproti základnímu nastavení regulátoru [logistická mapa].

| ⊞ | HC12 E\|TDAS | | | GE - Struk. | | | HC12 Tuning | | |
|---|---|---|---|---|---|---|---|---|---|
| | Nej. | MED. | STD | Nej. | MED. | STD | Nej. | MED. | STD |
| UPO1 | 16.989 | 17.940 | 3.702 | 6.064 | 6.716 | 13.909 | 4.487 | 4.866 | 12.896 |
| UPO2 | 30.378 | 32.371 | 5.776 | 26.192 | 36.423 | 7.204 | 12.043 | 31.421 | 13.461 |
| UPO4 | 37.955 | 40.766 | 3.185 | 31.364 | 38.834 | 6.139 | 21.663 | 35.398 | 9.720 |

**Tabulka 7.1:** Hodnoty výsledných fitness hodnot ze všech typů regulace [logistická mapa].

Kvalita stabilizace chaosového systému lze posoudit i schopností výsledného řešení konvergovat k žádané hodnotě. Grafy (graf: 7.6a) ETDAS, (graf: 7.6b) GE-Struk a (graf: 7.6c) GE-Tuning zobrazují průběh ustálení pro nejlepší nastavení metody pro 4 fixní body ze všech sekcí tohoto příkladu a v tabulce (tab: 7.2) jsou jednotlivé hodnoty konvergence. Hodnota konvergence je procentuální hodnota, kdy funkce konverguje v určitém požadovaném rozsahu.

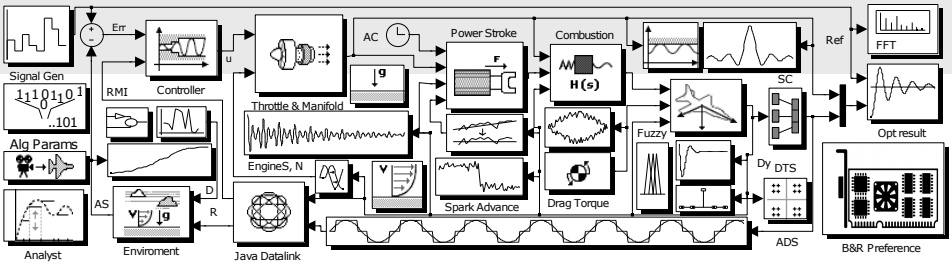


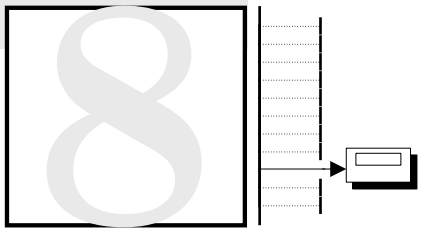



**Graf 7.6:** Grafy konvergence k žádané hodnotě pro UPO4 stabilizaci chaosu [logistická mapa]. Pořadí grafů: (a: ETDAS), (b: GE-Struk) a (c: GE-Tuning).

| ETDAS kongt | GE-Struk kongt | GE-Tuning kongt |
|---|---|---|
| 65.847 | 68.458 | 73.514 |

**Tabulka 7.2:** Hodnota konvergence k žádané hodnotě pro UPO4 stabilizaci chaosu [logistická mapa].



# KAPITOLA 8

# PŘÍKLAD STABILIZACE DUFFINGOVY ROVNICE

Stabilizace duffingovy rovnice je jako další příklad stabilizace tzv. deterministického chaosu, kdy systém je upravená duffingova mapa (vz: 8.1). Systém je stabilizovaný na dva fixní body (UPO) v hodnotách 0.85725557 a 1.40285623, které byly dosaženy na základe analytického výpočtu rovnice ze stabilizační sekvence (vz: 8.3), při počátečních podmínkách $x_0 = 0.1$ a $y_0 = 0.1$. Celková stabilizace systému je provedena na rozsah 600 iterací.

K nalezení optimálního stabilizátoru v tomto případě je rozděleno na dvě části, první část používá Pyragasovu metodu pro více fixních bodů (ETDAS) a druhá část je zaměřena na doladění hodnoty výsledku pomocí druhého chodu metody na základě speciálně upravených pravidel výběru kontrolované veličiny. Kdy doladění původní metody může přinést zkvalitnění výsledku už optimální stabilizační metody.

$$x_{n+1} = y_n, y_{n+1} = -bx_n + ay_n - y^3 \tag{8.1}$$

Fitness funkce, k ohodnocení parametru $x$ na základe iterace, je použitá stejná metoda jako v případě stabilizace logistické mapy, teda metoda (ITAE) s atraktorem hodnoty.

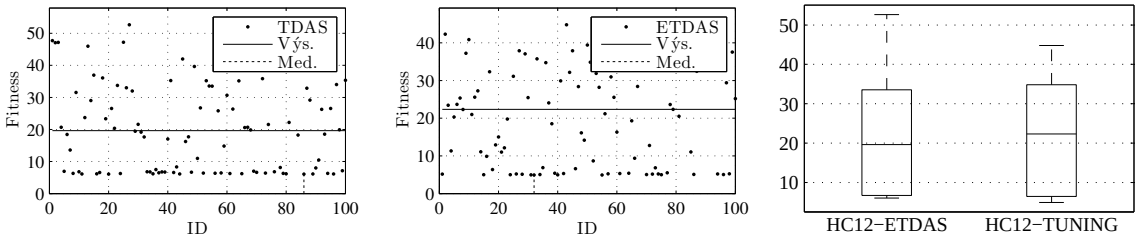
$$J = \begin{pmatrix} \frac{\partial P}{\partial x} & \frac{\partial P}{\partial y} \\ \frac{\partial Q}{\partial x} & \frac{\partial Q}{\partial y} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 - x^2 & -k \end{pmatrix} \tag{8.2}$$

Optimalizace parametrů obou metod hledání stabilizační metody je pomocí algoritmu (HC12), jako u předešlého přikladu, který představil vhodnost použití algoritmu na hledaní optimálního nastavení metody u deterministického chaosu.

$$\begin{aligned} F_1 &= 0.718039 - 0.451373x_{n-1} \\ F_2 &= 0.376078 - 0.329025x_{n-1} \end{aligned} \tag{8.3}$$

## 8.1 Zhodnocení výsledků

Zhodnocení výsledků pro předchozí dva příklady stabilizace duffingovy mapy. Statistika se provedla na algoritmus (HC12) při 100 opakování a pro každý příklad stabilizace. Jednotlivé výsledky jsou uvedeny v tabulce (tab: 8.1) s grafickým znázorněním nejlepší fitness hodnoty pro stabilizaci pomocí Pyragasovy metody (graf: 8.1a) a pro doladění metody pomocí sekundárního průběhu algoritmu (graf: 8.1b). Dle obou prezentovaných příkladů stabilizace duffingovy mapy lze najít optimální řešení problému a celková úspěšnost algoritmu je v box grafu (graf: 8.1c). Ke stabilizaci chaotického systému lze použít i (GE) algoritmus, kterému se podrobněji věnuje kapitola příklad stabilizace logistické mapy.

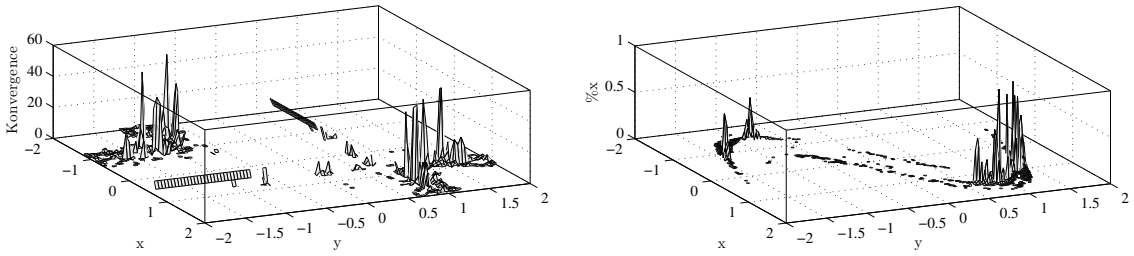



**Graf 8.1:** Hodnoty všech fitness hodnocení. Pořadí grafů: (a: HC12-ETDAS), (b: HC12-TUNING) a (c: Box graf hodnot fitness funkce pro všechny typy stabilizace [duffingova rovnice].).

| ⊞ | HC12-ETDAS | | | HC12-TUNING | | |
|---|---|---|---|---|---|---|
| | Nej. | MED. | STD | Nej. | MED. | STD |
| FIT | 6.064 | 19.648 | 14.119 | 4.937 | 22.333 | 13.325 |

**Tabulka 8.1:** Hodnoty výsledných fitness hodnot ze všech typů stabilizace [duffingova rovnice].

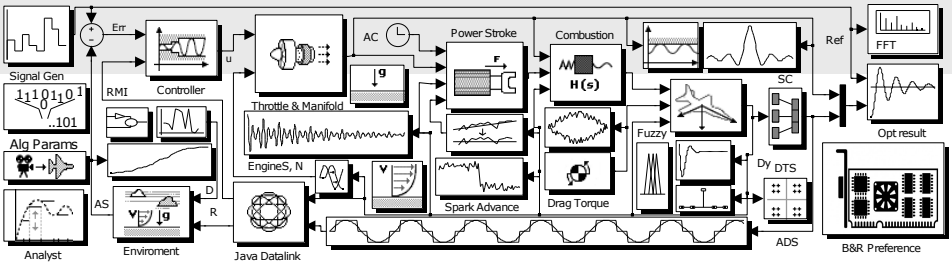
Kvalita hodnoty konvergence pro různé pozice fixních bodů se provedla na nejlepší výsledek stabilizace, (graf: 8.2a) prezentuje konvergence na celém rozsahu mapy $x, y < -2, 2 >$, rozdělené dle $Nx, Ny = 300$. Jednotlivé hodnoty jsou v tabulce (tab: 8.2), pro hodnoty původních hodnot (UPO), nejlepší nalezené hodnoty (UPO) a konvergenci celé mapy. Všechny hodnoty $x$ ze všech konvergujících možností nastavení (UPO) jsou v grafu (graf: 8.2b).

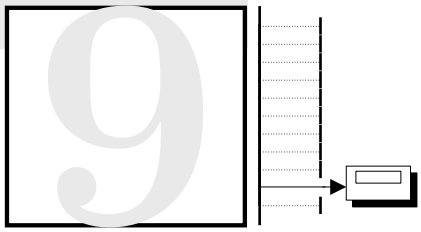



**Graf 8.2:** Grafy rozložení konvergence [duffingova rovnice]. Pořadí grafů: (a: Hodnota konvergence v závislosti na změně polohy UPO.) a (b: Hodnoty $x$ pro všechny konvergující polohy UPO.).

| Kongt | Alt. UPO1 | Alt. UPO2 | Alt kongt | Mapa kongt |
|---|---|---|---|---|
| 50.167 | -1.434 | -0.869 | 54.500 | 24.570 |

**Tabulka 8.2:** Hodnota rozložení konvergence při změně polohy UPO [duffingova rovnice].



# KAPITOLA 9

# PŘÍKLAD NÁVRHU ANTÉNY

Příklad návrhu Yagi-Uda antény pro co nejlepší parametry ve stanoveném frekvenčním pásmu 290 až 310 MHz. Toto pásmo bylo vybráno z toho důvodu, jelikož veškeré geometrické délky prvků antény jsou vztahovány k délce vlny $\lambda$. V případě frekvenčního pásma okolo 300 MHz je délka vlny zhruba rovna 1.

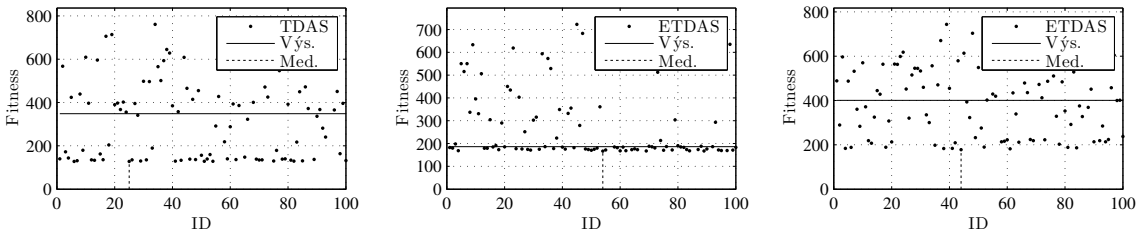
To nám dává lepší představu o celkových rozměrech navržené antény. Kvalita konstrukce je vyjádřena matematicky objektivní fitness funkcí. Tato funkce byla navržena experimentálně a zahrnuje tři vybrané parametry antény, které definují její celkovou kvalitu. Jelikož mezi jednotlivými parametry antény existuje jistá souvislost, takže výsledná anténa je vždy jakýmsi kompromisem mezi těmito parametry, kterými jsou zisk antény $G$, vstupní impedance antény vyjádřená pomocí poměru stojatých vln VSWR (zkratka vychází z anglického slovního spojení voltage standing wave ratio) a předozadní poměr $F/B$. Výsledná fitness funkce $F$ má tvar.

$$F = -G_{min} + (a * vswr_{max}) + \left(b * \frac{F}{B_{min}}\right) \tag{9.1}$$

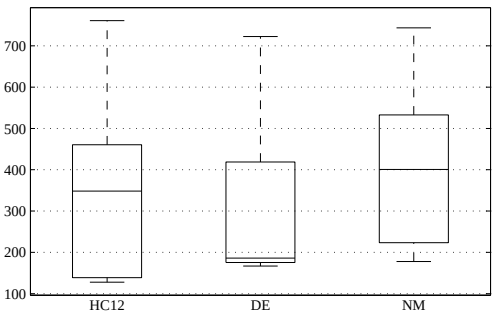
Pří návrhu antén je zvykem navrhovat anténu pouze pro střed uvažovaného pásma, což je v našem případě 300 MHz. Jelikož využíváme od samého počátku automatizovaného návrhu pomocí evolučních algoritmů, tak si můžeme dovolit strategii, kdy uvažujeme parametry v celé šířce pásma a z tohoto pásma jsme vybrali nejhorší variantu parametrů. Tento přístup omezil počet vyhovujících řešení, ale na druhou stranu se vyhneme situacím, kdy by výsledná anténa měla výborné výsledky pro střed pásma, ale pro okolní frekvence budou tyto parametry nevyhovující. V případě zisku a předozadního poměru uvažujeme jejich nejnižší hodnotu $G\ L$ a $F/B$ min. U poměru stojatých vln uvažujeme jeho nejvyšší hodnotu $vswr$ max. Pomocí konstant $a$ a $b$ můžeme následně upravit váhu, jakou budou mít parametry $vswr$ a $F/B$ vůči zisku $G\ L$. Konstanta $a$ určuje váhu poměru stojatých vln a nabývá těchto hodnot.

## 9.1 Zhodnocení výsledků

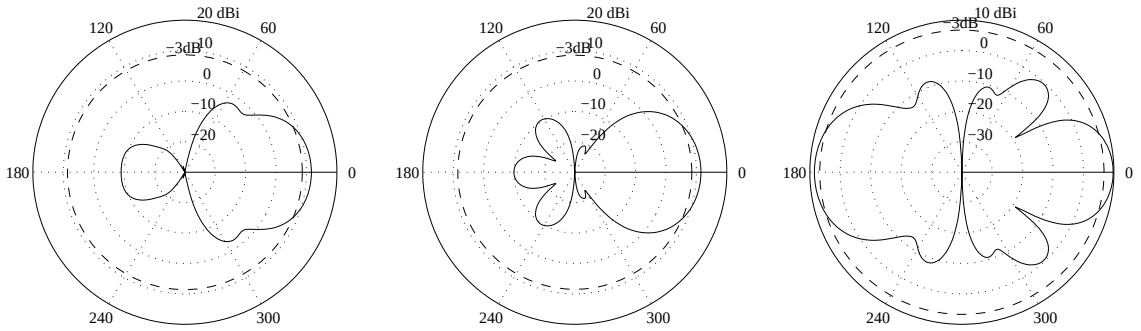
Na základě stanovené fitness funkce (vz: 9.1) byla provedena série simulací pro každý vybraný algoritmus. Nejlepší dosažené výsledky pro jednotlivé algoritmy jsou uvedeny v následující tabulce (tab: 9.1). Tabulka obsahuje i rozměry jednotlivých elementů antény a rozteče mezi jednotlivými elementy. Na konci tabulky jsou uvedeny dosažené hodnoty tří vybraných parametrů, které byly zahrnuty do fitness funkce. Průměrné hodnoty společně s extrémy od zisku po délku antény jsou uvedeny v grafech (graf: 9.3a), (graf: 9.3b) a (graf: 9.3c). Vyzařovací diagramy pro nejlepší nastavení antén dle jednotlivých algoritmů znázorňují grafy (graf: 9.2a), (graf: 9.2b) a (graf: 9.2c). Grafické znázornění jednotlivých antén je v grafech (graf: 9.4a), (graf: 9.4b) a (graf: 9.4c). Poslední porovnaní udává kvalitu jednotlivých algoritmů pro 100 opakovaní na jednom zadání dle grafu (graf: 9.1b), (graf: 9.1a) a (graf: 9.1c).



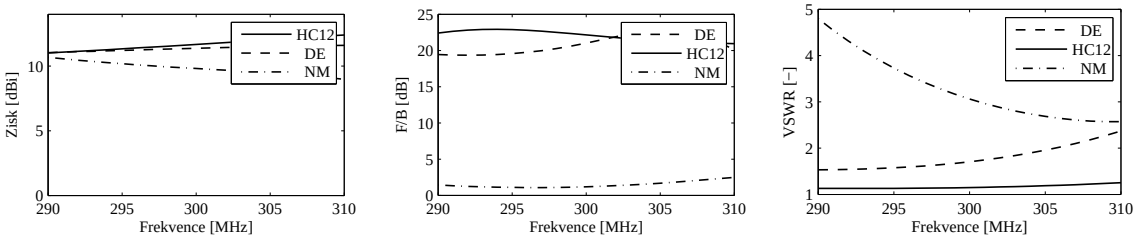

**Graf 9.1:** Hodnoty všech fitness hodnocení. Pořadí grafů: (a: HC12), (b: DE) a (c: NM).



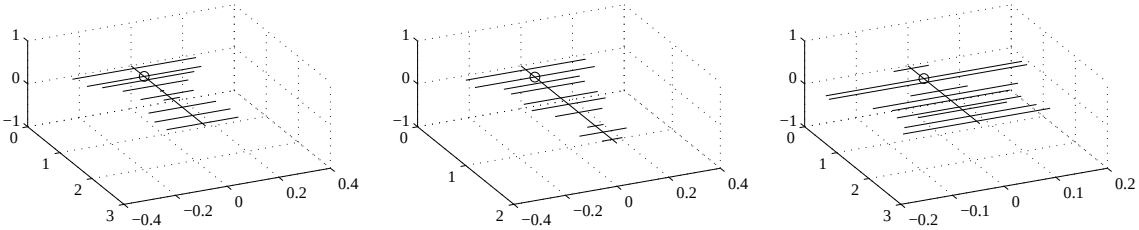

Návrh parametrů antény je komplexní problém skládající se z mnoha faktorů, které více či méně ovlivňují výsledný návrh antény. Mezi tyto faktory patří zejména směrová charakteristika antény, předozadní poměr, impedance a hlavně zisk. Všechny tyto funkce jsou vysoce nelineární. V tomto příkladu byl představen návrh Yagi-Uda antény. Na tomto konkrétním typu antény bylo demonstrováno použití algoritmů umělé inteligence. Byly využity algoritmy (NM), (DE) a (HC12). Fitness funkce (vz: 9.1) zahrnuje parametry zisku, předozadního poměru a poměru stojatých vln *vswr*. Veškeré simulace probíhaly v programu superNEC. Použití genetických algoritmů pro návrh parametrů antén je vhodné zejména tam, kde je potřeba návrh časově urychlit.




**Graf 9.2:** Vyzařovací diagramy. Pořadí grafů: (a: Návrh dle DE.), (b: Návrh dle HC12.) a (c: Návrh dle NM.).




**Graf 9.3:** Grafy průběhů. Pořadí grafů: (a: Průběh zisku.), (b: Průběh předozadního poměru.) a (c: Průběh poměru stojatých vln.).

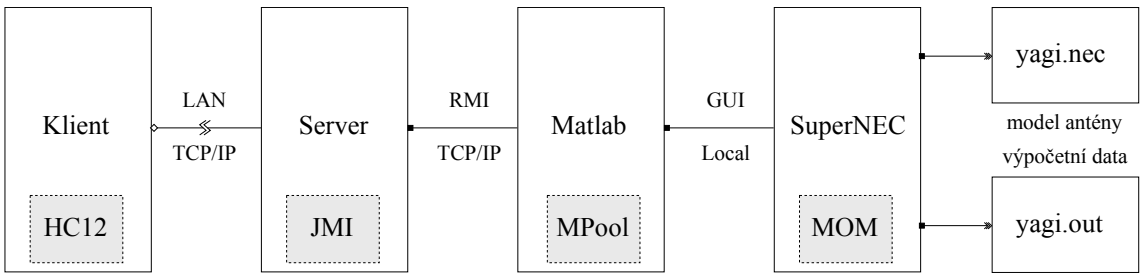


**Graf 9.4:** Návrh konečných antén. Pořadí grafů: (a: Návrh dle DE.), (b: Návrh dle HC12.) a (c: Návrh dle NM.).

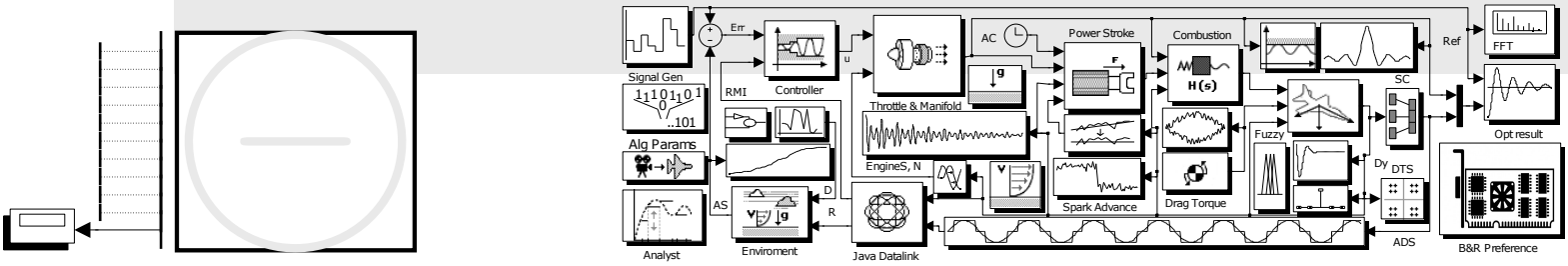
| Algoritmus | NM | | DE | | HC12 | |
|---|---|---|---|---|---|---|
| Element | Délka [m] $\lambda$ | Mezera [m]$\lambda$ | Délka | Mezera | Délka | Mezera |
| 1 | 0.068 | 0.395 | 0.479 | 0.310 | 0.464 | 0.190 |
| 2 | 0.380 | 0.130 | 0.443 | 0.126 | | 0.145 |
| 3 | 0.384 | 0.344 | 0.357 | 0.230 | 0.294 | 0.127 |
| 4 | 0.111 | 0.196 | 0.254 | 0.202 | 0.286 | 0.266 |
| 5 | 0.281 | 0.171 | 0.016 | 0.158 | | 0.164 |
| 6 | 0.228 | 0.258 | 0.206 | 0.145 | | 0.209 |
| 7 | 0.241 | 0.108 | 0.063 | 0.329 | 0.183 | 0.347 |
| 8 | 0.172 | 0.243 | 0.262 | 0.317 | 0.063 | 0.158 |
| 9 | 0.256 | 0.194 | 0.294 | 0.310 | 0.183 | 0.190 |
| 10 | 0.285 | - | 0.278 | - | 0.079 | - |
| Zisk [dBi] | 9.0 - 10.5 | | 11.0 - 12.5 | | 11.0 - 11.6 | |
| VSWR [-] | 2.55 - 4.95 | | 1.55 - 2.25 | | 1.13 - 1.25 | |
| F/B [dB] | 1.1 - 2.5 | | 19 - 23 | | 21 - 23 | |

**Tabulka 9.1:** Souhrn parametrů navržených antén.

SuperNEC je výpočetní program vyvinutý za účelem simulačního výpočtu antén a rozšiřuje dřívější verzi (NEC2). Program implementuje i paralelní výpočty na základě (PVM) nástrojů a umožňuje tak spuštění programu na více počítačích (operačních systémů), které jsou propojeny na základě standardního síťového protokolu TCP/IP.




**Obrázek 9.1:** Schéma simulačního programu na výpočet Yagi-Uda antén.




# ZÁVĚR

Problematika nelineárního řízení komplexních soustav a analýza výsledků je dosti složité téma, protože zanedbaní některých částí muže výrazně ovlivnit samotný výsledek. Proto je nutné ke každému problém přistupovat individuálně a pečlivě připravit a nastavit kvalifikační požadavky na požadovaný výsledek. V rámci této disertační práce byly prezentovány v problematice řízení komplexních soustav variance druhů regulace systémů od klasického matematického modelu až po reálný vysoce nelineární systém a reálný hardware. Dále byla práce rozšířena o nové postupy při stabilizaci nedeterministického chaosu a přikladu návrhu struktury antén.

V oboru regulace systémů je nutno postupovat dle platných předpisů právě díky závažným důsledkům, které by mohly pramenit ze špatné interpretace výsledku měření či dokonce nevhodně zvoleném postupu regulace.

Všechny zde uvedené příklady jsou porovnány se standartním způsobem řešení používaným v praxi a doplněny o nové poznatky s využitím evolučních algoritmů. Hodnocení výsledků je prezentováno na statistických grafech a ukazuje mnohé výhody využití tohoto postupu pro daný typ přikladu. Každá z kapitol v práci představuje jeden způsob řešení problému a představuje dále podpis daného postupu.

K dosažení všech výsledků bylo nutno vytvořit aplikační prostředky na kterých byly demonstrovány různé způsoby přístupů k dané problematice. Vytvořená aplikace splňuje mnohé výrazné vlastnosti a moderní přístupy k vývoji aplikací. Hlavní předností aplikace jsou způsoby zapojení dle server / klient konfigurace, multi plaformový design, využívání paralelních výpočtů a mnohé další vlastnosti prezentované ve vlastní kapitole.

Rozdělení práce je na teoretickou část, která se zabývá popisem problému v oblasti řízení systémů a optimalizace spolu s evolučními algoritmy, v kapitole 1 až 2, a praktickou část. Praktická část zahrnuje popis samostatné aplikace, kapitola 3, stručný popis všech řešených systémů spolu s návrhem hodnoticí funkce, kapitola 4. Kapitola 5 až 7 prezentuje postupy při regulaci komplexních soustav a kapitola 8 obsahuje zhodnoceni výsledků napříč všemi řešeními. Ke stabilizaci deterministického chaosu je věnována kapitola 9 až 10. Poslední kapitola představuje řešení návrhu obecných antén.

Práce nelineární řízení komplexních soustav s využitím evolučních algoritmů prezentuje mnohé nové přístupy a poznatky, které se dají dále rozvíjet a využít na reálných problémech v technické praxi. Celkové výsledky této práce se dají rozdělit na tyto dílčí výsledky:

- Obecně popsaná problematika regulace teorie řízení a optimalizace s popisem všech využívaných evolučních algoritmů v této práci.
- Systematický popis možného aplikačního přístupu.
- Aplikovaní problematiky do vlastního softwarového řešení.
- Využití znalostí multikriteriální optimalizace k návrhu vlastní hodnoticí funkce, zohledňující variabilitu vlastností při regulaci systémů.
- Provedení série testů na nové způsoby regulace systémů, a to konkrétně v rámci návrhu PID regulátoru.
- Využití Fuzzy logiky pro lepší návrh komplexního regulátoru.
- Postup vlastního návrhu struktury regulátoru a komplexní doladění s využitím evolučních algoritmů.
- Nové přístupy pro stabilizaci deterministického chaosu.
- Nové postupy a fitness funkce při navrhovaní struktury Yagi-Uda antény.
- Zpracovaní a analýza výsledků v grafech a statistických metodách.
- Porovnaní výsledků pro různé přístupy k řešení daného problému.
- Vytvoření samostatné aplikace umožňující další vývoj a použití v technické praxi.

QR metadata: (a)dokument, (b)PDF

# SEZNAM VLASTNÍCH PUBLIKACÍ

2017

*Zuth Daniel*, *Minář Petr*: *Aplikace fuzzy množin pro určení technického stavu stroje.* Diago 2017. Technická diagnostika, ročník. 26. z1 s. 87-92. ISSN: 1210- 311X.

2015

*Matyáš Pavel*, *Minář Petr*, *Ošmera Pavel*: *Design of Yagi- Uda antenna using HC12 algorithm.* In Mendel 2015. Mendel Journal series. 21th International Conference on Soft Computing. Mendel Journal series. 2. 2015. s. 143-149. ISBN: 978-3-319-19823-1. ISSN: 1803- 3814.

2014

*Matoušek Radomil*, *Dobrovský Ladislav*, *Minář Petr*, *Mouralova Kateřina*: *A Note about Robust Stabilization of Chaotic Hénon System Using Grammatical Evolution.* A Note about Robust Stabilization of Chaotic Hénon System Using Grammatical Evolution. Advances in Intelligent Systems and Computing, 2014, roč. 2014, č. 289, s. 219-228. ISSN: 2194- 5357.

*Matyáš Pavel*, *Minář Petr*: *Návrh Yagi- Uda antény pomocí evolučních algoritmů.* In Technical Computing Bratislava 2014. 2014. Slovakia, Bratislava: HUMUSOFT, 2014. s. 1-4. ISBN: 978-80-7080-898- 6.

2013

*Matoušek Radomil*, *Minář Petr*: *Stabilization of Chaotic Logistic Equation Using HC12 and Grammatical Evolution.* Advances in Intelligent Systems and Computing, 2013, roč. 2013, č. 210, s. 137-146. ISSN: 2194- 5357.

2012

*Matoušek Radomil*, *Minář Petr*, *Lang Stanislav*, *Pivoňka Petr*: *HC12: Efficient Method in Optimal PID Tuning.* Engineering Letters, 2012, roč. 20, č. 1, s. 42-48. ISSN: 1816- 093X.

2011

*Matoušek Radomil*, *Minář Petr*, *Lang Stanislav*, *Šeda Miloš*: *HC12: Efficient PID Controller Design*. Computational Engineering in Systems, IAASAT, 2011. s. 194-199. ISBN: 978-1-61804-026- 8.

*Matoušek Radomil*, *Minář Petr*, *Lang Stanislav*, *Pivoňka Petr*: *HC12: Efficient Method in Optimal PID Tuning*. Lecture Notes in Engineering and Computer Science, 2011, roč. 2011, č. 1, s. 463-468. ISSN: 2078- 0958.

*Matoušek Radomil*, *Minář Petr*, *Lang Stanislav*, *Pivoňka Petr*: *Evolutionary Design of Polynomial Controller*. An international Journal of Science, Engineering and Technology, World Academy of Science Engineering and Technology, 2011, roč. 2011, č. 59, s. 639-644. ISSN: 2010- 376X.

*Minář Petr, Matoušek Radomil CDF v1; Control Design Framework (PID, Fuzzy)*. VUT Brno. URL: http://www.uai.fme.vutbr. cz. (software).

2010

*Matoušek Radomil*, *Minář Petr*: *Optimalizace parametrů PSD regulátoru pomocí algoritmu HC12*. In Technical Computing Bratislava: RT Systems, 2010. s. 68-70. ISBN: 978-80-970519-0- 7.

2009

*Minář Petr*: *Distribuované výpočty s využitím technologie ActionScript*. Brno, 2009. Diplomová práce. Vyské učení technické v Brně.

*Minář Petr*: *Interaktivní webové aplikace vytvořené pomocí technologie Adobe Flash - Teorie Kódování*. Brno, 2009. VUT Brno. URL: http://www.uai.fme.vutbr. cz. (software).

# SEZNAM ZKRATEK A SYMBOLŮ

DE . . . . . Diferenciální evoluce

EA . . . . . Evoluční algoritmy
ETDAS . . . . . [ENG] Extended time delayed auto synchronization

FL . . . . . Fuzzy logika

GE . . . . . Gramatická evoluce

HC12 . . . . . Algoritmus HC12
HIL . . . . . [ENG] Hardware in the loop

IAE . . . . . [ENG] Integral of the absolute magnitue of error
ITAE . . . . . [ENG] Integral of time-multiplied absolute value of error

JMI . . . . . [ENG] Java to matlab interface

NEC2 . . . . . [ENG] Numerical Electromagnetic Code, Version 2
NM . . . . . Nelder-Mead simplexová metoda
NT . . . . . Neuronové sítě

PID . . . . . Proporcionální, integrační a derivační regulátor
PLC . . . . . [ENG] Programmable logic controller
PVM . . . . . [ENG] Parallel Virtual Machine

RMI . . . . . [ENG] Java remote method

SC . . . . . Soft computing
SU . . . . . Strojové učení

TDAS . . . . . [ENG] Time delayed auto synchronization

UPO . . . . . [ENG] Unstable periodic orbit

ZN . . . . . Ziegler-Nichols metoda nastavení regulátoru

# LITERATURA

[1] *Haupt L. Randy*, *Haupt Sue Ellen*: *Practical Genetic Algorithms*. Wiley-Interscience, 2004. ISBN 0471455652.

[2] *Holland, H. John*: *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. (1975) University of Michigan Press, Ann Arbor

[3] *Mathews H. John*, *Kurtis K. Fink*: *Numerical Methods Using Matlab*. Upper Saddle River, New Jersey, USA: Prentice-Hall Inc., 2004, s. 8. ISBN 0130652482.

[4] *Minář Petr*: *Distribuované výpočty s využitím technologie ActionScript*. Brno, 2009. Diplomová práce. Vyské učení technické v Brně.

[5] *Jin Y.*: *A Definition of Soft Computing - adapted from L.A. Zadeh* [online]. Computational intelligence, natural computing, and organic computing [online]. 2000 [cit. 2012-07-13] Dostupné z URL: <`www.soft-computing.de/def.html`>.

[6] *Tomek Eduard*: *Škálování implementace optimalizačního algoritmu na mnohajádrových strojích*. Brno, 2012. Bakalářská práce. Masarykova univerzita.

[7] *Zadeh A. Lotfi*: *Fuzzy sets*. (1965) Information and Control, 8 (3), pp. 338-353.

[8] *Sierra Kathy*, *Bates Bert*: *Head first Java*. Sebastopol, Calif.: O'Reilly, 2005. ISBN 0596009208.

[9] *Altman M. Yair*: *Undocumented Secrets of MATLAB-Java Programming*. London: Chapman and Hall/CRC, 2011. ISBN 1439869030.

[10] *May M. Robert*: *Simple mathematical models with very complicated dynamics*. Nature 261(5560):459-467. (1976)

[11] *Matoušek Radomil*: *Pokročilé metody počítačové intligence*. Brno, 2014. Habilitační práce. Vyské učení technické v Brně. ISBN 978-80.-214-4341-0

[12] *Calvert L. Kenneth*, *Donahoo J. Michael*: *TCP/IP Sockets in Java: Practical Guide for Programmers*. Morgan Kaufmann, 2008. ISBN 978-0123742551.

[13] *Grosso William*: *Java RMI*. O'Reilly Media, 2001. ISBN 978-1565924529.

[14] *Kepner Jeremy*: *Parallel MATLAB for Multicore and Multinode Computers*. SIAM-Society for Industrial and Applied Mathematics, 2009. ISBN 978-0898716733.

[15] *Fedorov V. Valerii*, *Leonov L. Sergei*: *Optimal Design for Nonlinear Response Models*. CRC Press, 2013. ISBN 978-1439821510.

[16] *B&R* : *Automation Studio Target for Simulink* [online]. Automation Studio tutorials [online]. Dostupné z URL: <`http://kyb.fei.tuke.sk/laboratoria/prezentacie/br%20automation%20studio%20target%20for%20simulink.pdf`>.

[17] *Ghaffari Azad*: *dSPACE and Real-Time Interface in Simulink* [online]. San Diego State University study materials [online]. Dostupné z URL: <`http://flyingv.ucsd.edu/azad/dSPACE_tutorial.pdf`>.

[18] *Minář Petr*: *Interaktivní webové aplikace vytvořené pomocí technologie Adobe Flash - Teorie Kódování*. Brno, 2007. Bakalářská práce. Vyské učení technické v Brně.

[19] *Pacheco Peter*: *An Introduction to Parallel Programming*. Morgan Kaufmann, 2013. ISBN 978-0123742605.

[20] *Linder Marek*, *Pernecký Daniel*, *Sekaj Ivan*: *Grid Computing in Matlab for Solving Evolutionary*. Technical Computing Bratislava 2012 [elektronický zdroj] : 20th Annual Conference Proceedings.

[21] *Sekaj Ivan*, *Linder Marek*: *Evolutionary Algorithm Based Power Plant Working Point Optimisation Using PLC and Simulink Model*. Technical Computing Bratislava 2010 : 18th Annual Conference Proceedings. Bratislava, Slovak Republic, 20.10.2010. - Bratislava : RT Systems, 2010. - ISBN 978-80-970519-0-7.

# CURRICULUM VITAE

## Osobní údaje


Jméno a příjmení: Petr Minář

Datum narození: 19.4 1984

Národnost: Česká

Kontakt: yminar08@stud.vutbr.cz


## Vzdělání

2009-2017 **PhD**, *Vysoké učení technické v Brně, Ústav automatizace a informatiky*, Brno, Česká Republika.
Doktorská práce: Nelineární řízení komplexních soustav s vyuľitím evolučních přístupů

2007-2009 **Ing**, *Vysoké učení technické v Brně, Fakulta strojního inľenýrství*, Brno, Česká Republika.
Diplomová práce: Distribuované výpočty s vyuľitím technologie ActionScript

2004-2007 **Bc**, *Vysoké učení technické v Brně, Fakulta strojního inľenýrství*, Brno, Česká Republika.
Bakalářská práce: Interaktivní webové aplikace vytvořené pomocí technologie Adobe Flash - Teorie Kódování

## Kariéra

2015-dosud **Software Design Engineer**, *EMS Satcom*, Tewkesbury, Velká Británie.
Vývoj vrstvy 2/3 z ISO/OSI modelu pro Inmarsat portfolio satelitní komunikace.

2009-2014 **Software Design Engineer**, *Honeywell*, Brno, Česká Republika.
Softwarový vývoj v aerospace, Datalink, Advance Technology Support.

2008 **Production Engineer Assistant**, *Maehler & Kaege CZ*, Brno, Česká Republika.
Vývoj softwarového základu pro produkční plánování.

## Ostatní dovednosti

- Elektrická vyhláška *(CZ edict.50/1978 cl.), §6*
- Certifikát: BR automation *(control, motion, visio)*
- Certifikát: Matlab (by Humusoft) *(s-function, parallel computing)*
- Certifikát: Six Sigma (by Honeywell) *(green belt design)*