Genetic Programming Enabled Evolution of Control Policies for Dynamic Stochastic Optimal Power Flow

Stephan Hutterer
School of Engineering and Environmental Sciences, Upper Austria University of Applied Sciences
Stelzhamerstrasse 23
Wels, Austria
stephan.hutterer@fh-wels.at

Stefan Vonolfen
School of Informatics, Communications and Media, Upper Austria University of Applied Sciences
Softwarepark 11
Hagenberg, Austria
stefan.vonolfen@fh-hagenberg.at

Michael Affenzeller
School of Informatics, Communications and Media, Upper Austria University of Applied Sciences
Softwarepark 11
Hagenberg, Austria
michael.affenzeller@fh-hagenberg.at

ABSTRACT

The optimal power flow (OPF) is one of the central optimization problems in power grid engineering, building an essential tool for numerous control as well as planning issues. Methods for solving the OPF that mainly treat steady-state situations have been studied extensively, ignoring uncertainties of system variables as well as their volatile behavior. While both the economical as well as well as technical importance of accurate control is high, especially for power flow control in dynamic and uncertain power systems, methods are needed that provide (near-) optimal actions quickly, eliminating issues on convergence speed or robustness of the optimization.

This paper shows an approximate policy-based control approach where optimal actions are derived from policies that are learned offline, but that later provide quick and accurate control actions in volatile situations. These policies are evolved using genetic programming, where multiple and interdependent policies are learned synchronously with simulation-based optimization. Finally, an approach is available for learning fast and robust power flow control policies suitable to highly dynamic power systems such as smart electric grids.

Categories and Subject Descriptors

I.2.8 [**Problem Solving, Control Methods, and Search**]: Heuristic methods

Keywords

Policy Learning, Simulation Optimization, Dynamic Stochastic Optimal Power Flow

GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.

1. INTRODUCTION

Operation of electric power grids, considered at an abstract level, traditionally happens according to "load-dependent generation". Here, the demand caused by (domestic, commercial and industrial) customers causes an electric load that can naturally not be influenced (in traditional power grid operation schemes), but can be predicted to a certain degree. Since the power-balance has to be maintained within a system, the generation has to be adapted continuously. This scheduling of power supply units is a central task in power grid operation [19].

In this context, several optimization problems have been formulated in the past [11], while the general optimal power flow (OPF) problem is probably the most important one. OPF aims at finding a configuration of all controllable supply units within a system in order to meet power demand at minimum financial costs, while maintaining several operational constraints. This OPF-based scheduling uses accurate demand forecasts in order to derive power flow control actions in a predictive manner.

However, especially in the context of smart electric grids, the volatility as well as uncertainty of power grid operations increases continuously while the implementation of distributed devices enlarges the amount of switchable appliances for power flow control. Thus, a scalable technology is needed for so called dynamic stochastic optimal power flow [10, 17]. Such a technology shall be proposed herein, where an evolutionary simulation optimization procedure is applied to learn flexible policies for power flow control, that later provide fast and robust actions at runtime.

To give an overview, Section 2 states the formulation and necessity of dynamic OPF, while discussing the approach of learning flexible control policies. Section 3 illustrates how such policies can be learned based on genetic programming, while this developed technique will be applied to a practical show case in Section 4. Finally, Section 5 provides concluding remarks.

2. DYNAMIC STOCHASTIC OPTIMAL POWER FLOW CONTROL

Before motivating the solution to dynamic OPF problems, for illustrating this optimization domain the reader shall be

supplied with the formulation of the common OPF problem which usually is applied to steady-state situations.

2.1 OPF Formulation

The standard OPF problem is defined to minimize the fuel cost per hour of a power system in steady state conditions, while fulfilling constraints for secure operation. The steady-state OPF problem can be mathematically formulated as follows [11, 19]:

$$min\ C(\overline{x}, \overline{u})$$

subject to:

$$g(\overline{x}, \overline{u}) = 0,$$
$$h(\overline{x}, \overline{u}) \leq 0,$$

where $C(\overline{x}, \overline{u})$ is the objective function for fuel cost minimization, $g(\overline{x}, \overline{u})$ is the set of equality constraints and gives typical load flow equations, $h(\overline{x}, \overline{u})$ provide the (inequality) system operation constraints. The inequality constraints $h(\overline{x}, \overline{u})$ represent the limits on physical power system equipment as well as the limits created to ensure system security. $\overline{x}$ and $\overline{u}$ reflect the vectors of dependent and independent variables. Hence, $\overline{x}$ gives the vector of (dependent) system variables: slack bus real power output P_{G_1}, load bus voltages V_L, generator reactive power outputs Q_G and branch power flows P_B.
The vector $\overline{u}$ comprises (independent) control variables: generator voltages V_G, generator real power outputs P_G, transformer tap settings T and the output of shunt VAR compensators Q_C. Hence, both vectors are expressed as:

$$x^T = [P_{G_1}, V_{L_1} ... V_{L_{NL}}, Q_{G_1} ... Q_{G_{NG}}, P_{B_1} ... P_{B_{NB}}]$$
$$u^T = [V_{G_1} ... V_{G_{NG}}, P_{G_2} ... P_{G_{NG}}, T_1 ... T_{NT}, Q_{C_1} ... Q_{C_{NC}}],$$

with NL, NG, NB, NT and NC reflecting the number of load buses, generator buses, branches, transformers and shunt VAR compensators respectively.
The objective function is specified to minimize the total fuel costs per hour over all generators. The specific costs C_n of each individual unit are expressed typically by a polynomial of degree 2:

$$C(\overline{x}, \overline{u}) = \sum_{n=1}^{NG} C_n = \sum_{n=1}^{NG} (a_n + b_n * P_{G_n} + c_n * P_{G_n}^2),$$

with a_n, b_n and c_n representing each generation unit's cost coefficients.
While minimizing the objective function, the following inequality constraints have to be satisfied, considering generation capacities

$$P_{G_n}^{min} \leq P_{G_n} \leq P_{G_n}^{max}, \quad (1)$$
$$Q_{G_n}^{min} \leq Q_{G_n} \leq Q_{G_n}^{max}, \quad (2)$$

as well as the voltage deviation being restricted to

$$V_j^{min} \leq V_j \leq V_j^{max} \quad (3)$$

over all buses $j = 1, ..., J$ and all generators $n = 1, ..., NG$. Branch flows need to be constrained to

$$P_b \leq P_b^{max} \quad (4)$$

for all branches $b = 1, ..., NB$ for enabling secure distribution grid operation.

Equations 1 - 4 define standard load flow constraints in power grids that have to be satisfied for enabling valid and secure power flow operation.

Additionally, the control variables for transformer tap settings and VAR compensators need to be restricted to the ranges:

$$T_{nt}^{min} \leq T_{nt} \leq T_{nt}^{max}, \quad (5)$$
$$Q_{C_{nc}}^{min} \leq Q_{C_{nc}} \leq Q_{C_{nc}}^{max}, \quad (6)$$

for all transformers $nt = 1...NT$ and VAR compensators $nc = 1...NC$.
The variable $Q_{C_{nc}}$ represents the reactive power injection from the shunt-connected static reactive power (VAR) compensators. These control-devices are applied for managing real-power balance, hence, are related to voltage regulation. The variable T_{nt} gives the tap position for transformers with tap-changing mechanisms. This technology allows the secondary-side adaptation of the actual number of used turns along a winding, enabling voltage regulation at the transformer output.

The traditional OPF formulation is about finding a steady-state solution $\overline{u}$ for a given situation, thus, determining concrete values for all control variables. However, in dynamic situations it would be more appropriate to have flexible control actions on hand that guarantee (near-) optimal power flow control, rather than static values for control variables.

2.2 Necessity of Dynamic OPF

The solution of this OPF problem addresses exactly one stationary state $S(t)$, disregarding possible states in the near future or possible uncertain conditions in the system. Considering the system exemplarily one time step later ($S(t+1)$) due to changing conditions of weather, customer-behavior or any other influence, the power flow in the system would change, hence, requiring a new solution to the optimal power flow problem further necessitated by the non-linear behavior of an electric power distribution system.
Such a new computation would need a robust and fast-converging solution method, that guarantees quick support with a new (near-) optimal solution, independent of system complexity and starting point, which cannot be guaranteed by traditional steady-state OPF methods [16]. However, this forms a challenge to dynamic optimization methods.

2.2.1 Power Flow Control in Smart Grids

The necessity of providing dynamic OPF methods is further substantiated by the steady increase of implementations of smart grid applications. Here, the aim is to control a huge amount of distributed devices (controllable loads, plug-in electric vehicle charging infrastructures, distributed storages, small-scale renewable generation units) that mostly show uncertain behavior. Thus, power flow control actions need to be provided for numerous appliances (i.e. high amount of control variables), that are fast in order to react to volatile situations, enable the incorporation of uncer-

tain behavior and are scalable to high amounts of devices. Thus, a scalable technique is needed for dynamic stochastic optimal power flow (DSOPF) control [10, 17, 18].

2.3 Policy-Based Dynamic OPF

The basic idea is to not re-optimize power flow control actions each time the system changes (for example because of changing weather or customer demand conditions), but to provide some kind of function $p(x)$ that supplies (near-) optimal control actions to any arbitrary situation that may occur, a so called policy. In actual approaches, this issue is about value- or policy- function approximation [14], where one has to assume that the structure of $p(x)$ is obvious or even can be obtained in some way.

Tentatively, we assume a fixed mathematical structure $p(\overline{i})$ that represents the policy, where the vector $\overline{i}$ contains all relevant input variables for decision making. This policy $p(\overline{i})$ serves as optimal power flow controller and provides the controllable unit respectively the grid operator with fast and robust actions.
In order to determine such a policy, one has to discuss the available variables that would need to be taken into account when aiming at deriving valid power flow control decisions. Considering the general OPF-problem as stated above, these input variables $\overline{i}$ would have to comprise the following information:

- Real and reactive load values of all buses; $P_{B_1}...P_{B_J}, Q_{B_1}...Q_{B_J}$,
- Real and reactive generation limits of all generators; $P^{max}_{G_1}...P^{max}_{G_{NG}}$, $Q^{max}_{G_1}...Q^{max}_{G_{NG}}$,
- Polynomial cost coefficients of all generators; $a_1...a_{NG}, b_1...b_{NG}, c_1...c_{NG}$,
- Real power flow limits of all branches; $P^{max}_{b_1}...P^{max}_{b_{NB}}$,
- Voltage deviation limits of all buses; $V^{min}_{B_1}...V^{min}_{B_J}, V^{max}_{B_1}...V^{max}_{B_J}$,
- Transformer tap setting limits; $T^{min}_1...T^{min}_{NT}$, $T^{max}_1...T^{max}_{NT}$,
- VAR compensators injection limits; $Q^{min}_{C_1}...Q^{min}_{C_{NC}}$, $Q^{max}_{C_1}...Q^{max}_{C_{NC}}$.

All these variables need to be considered if one aims at deriving a power flow control action that satisfies the OPF formulation.

A control action (i.e. desired output of a policy) gives control values to independent variables. Since the controllable variables for OPF as defined above are:

$$u^T = [V_{G_1}...V_{G_{NG}}, P_{G_2}...P_{G_{NG}}, T_1...T_{NT}, Q_{C_1}...Q_{C_{NC}}],$$

for each of these variables one policy $p(\overline{i})$ would have to be learned, hence, a number of $|\overline{u}| = 2 * NG + NT + NC$ policies would be needed for policy-based dynamic (approximate) optimal power flow control in this case.
While all these variables/policies are indeed interrelated, they would need to be evolved synchronously in order to maintain their interdependencies, making this optimization process a hard task from a computational point of view because of the high number of interrelated solutions. Further, the number of policies in this case directly grows with the size of the considered power grid (size in means of: number of generators NG, number of transformers NT, etc.; i.e. number of controllable units) which clearly builds a conflict with the requirement of building a scalable technology for high amounts of controllable devices in smart grid environments. Thus, an approach would be needed, where the number of needed policies is independent of the power grid's size. For example it should be possible to derive a general policy $P_G(\overline{i})$ that is valid for all generators within a system and is able of considering each generator's specific situation. This can be achieve when using so called "abstract rules".

2.4 Formulation of Abstract Rules

The approach of applying such abstract rules for evolving flexible policies has already been discussed in various applications for smart grid control tasks [6, 7]. The aim of these rules is to provide situation-dependent information to controllable units, while gathering unit-specific information from various needed system variables in a cumulated manner. Hence, for instance a generation unit may use the absrtact rule NLF (Neighborhood Load Factor) for considering the demand conditions within the neighbored area instead of taking all the concrete active power load values of related buses into account. Table 1 shows the set $\overline{r}$ of specified rules for OPF control. The rightmost column indicates to which control variable a respective rule is important. Finally, this set of rules substitutes the system variables, thus all the needed information for decision making is provided by abstract rules.

When finally synthesizing control policies out of these rules, the great advantage is achieved that such a policy is completely generic since it depends on abstract information. Hence, for all units in a system that perform control actions on for instance real power injection (variable P_G), only one policy $P_G(\overline{r})$ needs to be learned. Since this policy takes local information through its abstract rules, it derives unit-specific actions tailored to the controllable unit's individual environment and needs. In the end, for policy-based OPF control (related to above OPF definition) with abstract rule synthesis only 4 policies have to be evolved, namely $P_G(\overline{r})$, $V_G(\overline{r})$, $T(\overline{r})$ and $Q_C(\overline{r})$ respectively. This number is independent of the considered system's size, thus, this approach fulfills the scalability requirement.

3. GENETIC PROGRAMMING ENABLED SIMULATION-BASED EVOLUTION OF POLICIES

At this point, the kind of information that OPF control policies have to consider has been specified. Still, it is an open issue how to evolve these policies such that they lead to near-optimal control actions. In this work, genetic programming is applied for synthesizing the final policies $p(\overline{r})$ out of abstract rules $\overline{r}$. Since this synthesis is an optimization problem where the aim is to derive performant policies that output fast and robust control actions under uncertain conditions, a simulation-based learning procedure is applied similar to [6, 7].

Table 1: List of Abstract Rules

Rule	Explanation	Variable
LLF	Local Load Factor: active load at bus divided by maximum active power output at bus	P
NLF	Neighborhood Load Factor: sum of active load at directly connected buses and their neighbors divided by maximum active power output at those buses	P,V,Q,T
GLF	Global Load Factor: sum of total active load in grid divided by sum of maximum active power generation	P,V,Q,T
MARF	Max Rating Factor: maximum MVAR rating of connected branches divided by maximum MVAR rating of all branches	P,Q
MERF	Mean Rating Factor: mean MVAR rating of connected branches divided by maximum MVAR rating of all branches	P,Q
LCF	Linear Cost Coefficient: linear cost coefficient of generator divided by maximum linear cost coefficient of all generators	P
QCF	Quadratic Cost Coefficient: quadratic cost coefficient of generator divided by maximum quadratic cost coefficient of all generators	P
NRLF	Neighboring Reactive Load Factor: sum of reactive load at directly connected buses and their neighbors divided by maximum reactive power output at those buses	V,Q,T
GRLF	Global Reactive Load Factor: sum of total reactive load in grid divided by sum of maximum reactive power output	V,Q,T

3.1 Introduction to Genetic Programming

Genetic programmin (GP) uses an evolutionary-inspired heuristic search process for evolving computer programs of manifold style [1, 9] (rather than any kind of binary/integer/ real-valued vector like standard genetic algorithms). Within this work, such a computer program takes the appearance of a structured tree, where terminal nodes represent rules as defined before, that are combined together with arbitrary constants by a set of mathematical operators which are incorporated by non-terminal nodes. The set of these applied mathematical operators builds the GP grammar. This kind of solution representation allows arbitrary mathematical combination of abstract rules and thus is able of identifying even complex (non-linear) coherences between rules.

3.2 Policy Synthesis

Figure 1 gives an exemplary GP solution (tree) that may represent a policy for the control variable P_G, i.e. $P_G(\overline{r})$. In this case, the applied grammar consists of arithmetic functions. Here, non-terminal nodes representing mathematical operators are indicated in dashed style, while terminal nodes containing either abstract rules or real-valued constants are given in solid style. In this case, the policy would take the load situation at neighboring buses (NLF), the global load situation (GLF) as well as the linear costs of the considered generation unit (LCF) in order to derive the real-valued control action on P_G - i.e. the real power injection value of the generator. Once more, all controllable units of same type receive the same policy, but since the policy takes unit-specific information at runtime, individual control actions for each unit (like for generator real-power injection in this case) are provided.

3.3 Simulation-Based Evolutionary Optimization

For learning GP policies, simulation-based optimization according to [8] is applied for handling this problem. The

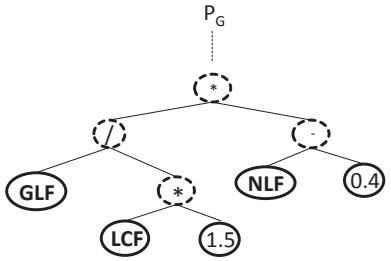

Figure 1: Exemplary Control Policy for Variable P_G

central idea of this approach is the application of simulation for evaluating the fitness of a solution candidate generated by the optimization algorithm. The advantage of using simulation for solution evaluation is manifold [4]. In this work, simulation is necessary to model the dynamic behavior of the considered power system along a considered time interval. Thus, with simulation the performance of policies is not only evaluated with respect to a steady-state situation $S(t)$, but is simulated along different dynamic system states $S(t)...S(t+K)$. Further, with simulation it is possible to integrate probabilistic influences into the model, which is important for considering real-world uncertain conditions in power flow control, enabling the evolution of not only accurate but also robust control policies.

For modeling the power flow control in a dynamic and uncertain environment based on a distribution grid model, the power grid's volatile states are simulated using MATPOWER [20] along a time-horizon of $K = 96$ discrete time steps ($\Delta k = 15$ minutes, i.e. time horizon equals one day) in order to create a dynamic environment. Here the demand changes over time according to statistical profiles from power grid operation, but is additionally randomized in order to simulate real-world uncertain conditions. In each time-step,

steady-state power flow computation is performed for simulating the power grid's response to certain policies' actions, where the stochastic variables (i.e. load values) are sampled from respective distributions. With this power flow computation, in each state the objective function value as well as the constraints satisfaction can be evaluated.

3.3.1 Formal Description

Within each simulated discrete time step, standard steady-state security constraints as defined in the general OPF problem (see Equations 1 - 6) need to be satisfied, the objective function shall be specified of minimizing financial costs of policy-controlled power generation C_p over all NG generation units during all K simulated time steps:

$$min \sum_{t=1}^{K} \sum_{n=1}^{NG} C_p(P_{G_n,t}, V_{G,t}, Q_t, T_t)$$

Since a dynamic case is considered where the power injection at a generation site may vary along time, additional ramping constraints need to be defined for variable P_G in order to ensure that this variation is within certain physical limits. Thus,

$$\forall n \forall t : |P_{G_n}(t+1) - P_{G_n}(t)| \leq \Delta P_{G_{n,max}} \quad (7)$$

Considering proportional penalization for constraint violation, the final fitness function is defined as:

$$min \sum_{t=1}^{K} \sum_{n=1}^{NG} C_p(P_{G,t}, V_{G,t}, Q_t, T_t) + \overline{w} * \overline{CV(t)}], \quad (8)$$

where the cardinality of $\overline{w}$ equals the number of considered constraints. Here, $\overline{w}$ contains each constraint's weight with respect to the objective function. The constraint violation $CV(t)$ is the quadratic error of a constrained variable exceeding the defined limits. The real power injections of generators P_{G_n} implicitly result from the policies' outputs after power flow simulation in each time step.

3.3.2 Training & Test Scenarios

In order to make the policies robust to uncertain situations, the evaluation has to be performed within a volatile as well as uncertain simulated environment. Therefore, in each evaluation a load-profile is chosen randomly out of four different profiles which finally describes the power-grids behavior along the simulated day. These profiles are given in Figure 2, building domestic (black solid), commercial (dashed) as well as agricultural (grey solid) standard load profiles. Additionally, a random load profile indicated by the grey dotted line is generated that is only used for testing-reasons lateron. Within each time step, the concrete load value at each node in the system is additionally randomized by multiplying it with a random sample from $N(0, 0.016)$, which is a common assumption for demand prediction uncertainty.

For better understanding, this workflow when evaluating a solution (i.e. set of policies) is illustrated in Figure 3.

After having optimized the policies, a proper test scenario is generated for validating the best found solution on an arbitrary volatile environment of the considered power grid.

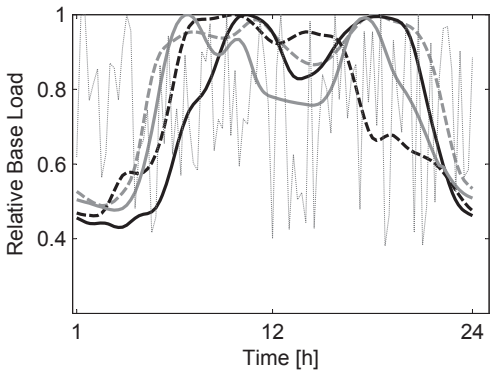

Figure 2: Applied Demand Profiles for Distribution Grid Simulation

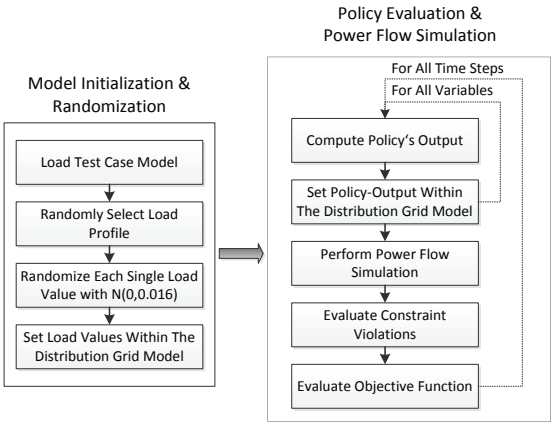

Figure 3: Solution Evaluation Procedure

Therefore, the synthetically generated random-profile (dotted grey line in Figure 2) is applied to the power grid model, creating a completely new situation independent from the load profiles that have been taken for training. Thus, the policies' robustness to highly volatile situations can be tested.

3.4 Evolution of Policies

Considering the problem formulation with special respect to the fitness function (Equation 8), the issue arises that this fitness depends on four policies $(P_{G,t}, V_{G,t}, Q_t, T_t)$, while these policies are indeed interrelated (for instance its obvious, that the voltage control performed by the tap changer T depends on the voltage control V_G of the generator units). Thus, it is not sufficient to evolve these policies one-by-one, but they have to be evolved all at a time. Hence, a coevolution-related scheme is necessary.

Coevolution is often essential when considering real-world problems. Such problems naturally consist of distributed entities that have own behavior but a global goal needs to be achieved as result of their interactions, hence, coevolution is strongly related to multi-agent systems [5, 13]. Here, coevo-

lutionary algorithms have emerged that use parallelization techniques for evolving disjunct subpopulations of different species, such as with coevolutionary genetic algorithms.
Originally, in evolutionary computation two main parallelization schemes have emerged, namely island models and diffusion models [1]. However, these models evolve populations of a single species, while individuals of this species exist within different subpopulations. Contrary, the aim of coevolutionary GA is to realize the coexistence of several species (in this case different control variables) that aim at a common goal (such as minimization of generation costs). Another distinction has to be made between cooperative and competitive coevolution [12, 15], while in this case clearly a cooperative scheme lies on hand.

3.5 Proposing Coevolutionary Genetic Programming

Numerous approaches for coevolutionary genetic algorithms have been developed in recent years, many of them realizing multi-agent or game-theoretic approaches for matching parallelized populations [3]. Within this work, a method is shown using a more straight-forward coevolutionary scheme, where a global fitness function is shared along parallely executed genetic programming processes. The idea is that n subpopulations $X_1 ... X_n$ are evolved within separate processes of same population size i, where evolutionary operations (mutation, parent selection, recombination) are applied separately and independently from each other. Each process evolves one of n policies using its individually defined grammar. The only information that the processes share is the common fitness of individuals belonging to the same solution. Thus, a complete solution X consists of n partial solutions (policies), i.e. $X = \{X_1, X_2, ...X_n\}$. This principle is shown Figure 4. When evaluating a solution candidate X, all policies that belong to X serve as input to the simulation model. After computing the respective fitness function value, it is shared along all partial solutions.
While all other genetic operators are executed locally, survivor selection happens globally. Being realized by a proportional selection scheme, complete solutions (such as a row $\{X_{1.1}, X_{2.1}, ..., X_{n.1}\}$ in Figure 4) are selected for replacement, rather than making this selection within each process independently. This is important regarding the nature of coevolutionary systems, where the fitness of a partial solution is always an objective (global) measure depending on other partial solutions, rather than a subjective measure.

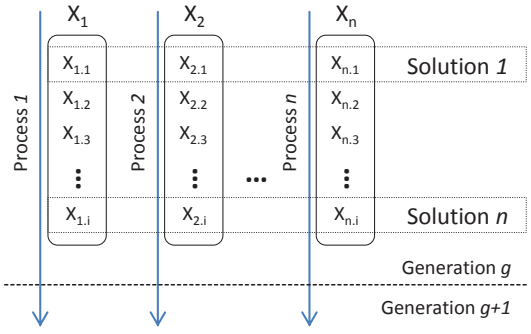

Figure 4: Process Model for Coevolutionary GP

For realization, HeuristicLab[1] is chosen as framework for heuristic optimization offering sophisticated implementations for genetic programming. Here, n separate processes are started where fitness values are shared using MPI (Message Passing Interface) in Windows OS.

4. EXPERIMENTS

For experimental validation, a dynamic stochastic OPF shall be handled for the IEEE 14-bus test case. Model data are taken from IEEE test case archive [2]. The model is implemented in MATPOWER simulation toolbox. The technical outline of this model is given in Figure 5, where buses with controllable units are annotated as follows: G ... Generator (variables P,V), Q ... VAR Compensator, T ... Tap Changer

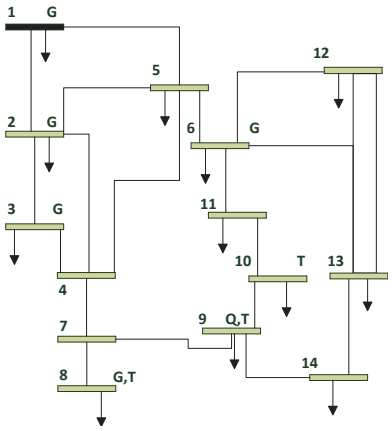

Figure 5: Test Case Layout; G ... Generator, Q ... VAR Compensator, T ... Tap Changer

For the genetic programming processes, the following parameter settings are applied as listed in Table 2. Further details on these parameters can be obtained from literature [1] respectively HeuristicLab.

Parameter	Value
Maximum Generations	200
Population Size	400
Selector	Proportional Selection
Mutation Probability [%]	15
Mutator	Multi Symbolic Expression Tree Manipulator [2]
Crossover	Subtree Swapping Crossover
Elitism	No Elitism
Maximum Tree Depth	8
Maximum Tree Length	80
Tree Grammar	Arithmetic Operators Real-Valued Constants

Table 2: GP Parameters: All Test Cases

For testing the best found policies, a proper test environment is applied as discussed before, where the power grid

[1] http://dev.heuristiclab.com/
[2] Randomly choose from: Full Tree Shaker, One Point Shaker, Remove Branch Manipulation, Replace Branch Manipulation

is simulated with an arbitrary volatile load profile over the considered 24 hours. In order to make the results comparable, out of this simulation 10 discrete time steps (i.e. discrete states) are selected. Within these time steps, the exact steady-state OPF is solved in MATPOWER with interior point solver. Than, the resulting objective function value (i.e. fuel costs per hour) is compared to the objective function value that results from the best found policies' actions within these 10 test- states.

4.1 Results

The evaluation on the described test set is shown in Table 3. For each state, the policies' objective function value and interior point solution's objective function value are indicated as well as their difference (relative error).

Time Step	Flexible Policy	Interior Point	Relative Error
1	8211.4	8151.5	0.0074
2	7288.2	7249.5	0.0053
3	4301.6	4256.8	0.0105
4	5157.2	5134.3	0.0045
5	5268.2	5246.5	0.0041
6	8020.7	7964.1	0.0071
7	8020.7	7964.1	0.0071
8	7732.2	7683.2	0.0064
9	6012.4	5991.8	0.0034
10	6705.8	6677.8	0.0042
Mean Relative Error			**0.0060**

Table 3: Results 14-Bus Dynamic OPF Control

From these results one can clearly derive, that the approximate optimal actions that the policies deliver within each situation are competitive with those that an exact OPF solution provides when being re-optimized within each state, where the mean relative error is only 0.6%.

At this point, it is important to mention that only 4 policies have been learned here for in sum 14 control variables (5 generators with controls P,V, 3 transformer taps and 1 VAR compensator). If another distribution grid would be considered with for example a higher number of controllable generation units, still only 4 policies would be needed for learning dynamic OPF control policies on this power grid. Thus, the technology of learning flexible control policies with abstract rules is highly scalable to applications with lots of distributed control devices. Especially the synthesis with GP allows the identification of complex nonlinear relationships between the abstract rules, enabling the optimization of powerful control policies. Considering future applications, for example control policies could be evolved for high amounts of controllable plug-in electric vehicles when charging their batteries like in [7], or dynamic charging/discharging policies for distributed small-scale storages.

While Table 3 summerizes the performance of the achieved solutions, the outlook of these best found policies should be discussed as well:
Figure 6 gives the exemplary tree representation of the final policy $P_G(\overline{r})$ for real power injection, while all 4 policies are listed as algebraic equations. For making the equations more readable, the acronyms of abstract rules are substituted by their indices $r_1 ... r_9$ in the vector if rules $\overline{r}$. This assignment is given in Table 4. The respective constants within the policies are rounded in order to make them more readable.

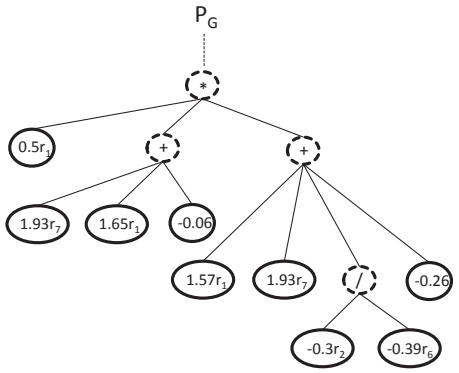

Figure 6: Policy $P_G(\overline{r})$ for Real Power Injection

$$P_G(\overline{r}) = 0.5r_1(1.93r_7 + 1.65r_1 - 0.06)* \\ (1.57r_1 + 1.93r_7 + \frac{-0.3r_2}{0.39r_6} - 0.26)$$

$$V_G(\overline{r}) = \frac{r_8(1.763r_8 - 0.328r_9) * 36.11}{(1.81r_8 - 0.32r_9)(-17.84r_8 + 106.31)} \\ + 0.92$$

$$Q(\overline{r}) = \frac{0.087r_5^3}{r_3r_4r_8(0.36r_5^2r_3 - 13.04r_2)} + 1.28r_4 \\ + 2r_8 - 4.34r_2$$

$$T(\overline{r}) = 0.65r_8$$

Variable	Rule
r_1	LLF
r_2	NLF
r_3	GLF
r_4	MARF
r_5	MERF
r_6	LCF
r_7	QCF
r_8	NRLF
r_9	GRLF

Table 4: Variables Assignment

Taking a look at $P_G(\overline{r})$, this policy takes the rules LLF and NLF as needed information on the actual load situation in the grid, and additionally the cost information about the generators, thus LCF and QCF. It seems that all other information that is needed for making a valid action on the real power injection can be gathered within the used constants, avoiding additional usage of abstract rules.
The same holds for the other policies as well, that do not need to take all provided information into account through the whole vector $\overline{r}$, but only need a subset of all rules. This is a special ability of genetic programming, which performs an implicit feature selection during the genetic search, which enables evolving solutions of low complexity. This low complexity is further enabled by using abstract rules instead of

system variables, which gather the needed information for making power flow decisions into single scalar values.

5. CONCLUSIONS

Future power flow control tasks in smart electric grids require optimization methods that are capable of deriving fast and robust control actions in dynamic and uncertain environments for potentially high amounts of controllable devices. Thus, the need for scalable technologies enabling dynamic stochastic optimal power flows is fundamental.
This work proposes an approach for dynamic approximate optimization, where flexible control policies are learned offline that later provide (near-) optimal control actions at runtime. One special ability of this approach is that respective policies are learned out of abstract information entities - so called abstract rules - that make it highly scalable. The learning procedure is realized using genetic programming, where the synchronous optimization of multiple interrelated policies is implemented using a coevolution-related scheme. Since uncertainties of a complex power grid have to be taken into account for evolving valid policies for real-world optimization, simulation is applied as system representation that allows to fully integrate the probabilistic system behavior into the optimization process.
Finally, the proposed technique has been applied to a well-know benchmark system, the IEEE 14-bus test case. Out of this benchmark, a dynamic power system has been simulated for evolving policies. It has been shown, that for an arbitrary set of discrete steady-state test states, the policies' outputs lead to competitive power flow control actions when comparing it to statically optimized exact solutions within these states. Thus, a technology is available for accurate and scalable dynamic stochastic optimal power flow control.

6. ACKNOWLEDGMENTS

The work described in this paper was done within the Josef Ressel Centre for Heuristic Optimization Heureka! (http://heureka.heuristiclab.com/) sponsored by the Austrian Research Promotion Agency (FFG).

7. REFERENCES

[1] M. Affenzeller, S. Wagner, S. Winkler, and A. Beham. *Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications*. CRC Press, 2009.

[2] R. D. Christie. Power systems test case archive; http://www.ee.washington.edu/research/pstca/, Mar. 2013.

[3] S. Ficici and J. Pollack. A game-theoretic approach to the simple coevolutionary algorithm. In *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature (PPSN VI)*. Springer-Verlag, 2000.

[4] M. C. Fu. Feature article: Optimization for simulation: Theory vs. practice. *INFORMS Journal on Computing*, 14, 1977.

[5] D. Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena*, 42(1-3), 1990.

[6] S. Hutterer, M. Affenzeller, and F. Auinger. Evolutionary algorithm based control policies for flexible optimal power flow over time. In *Proceedings of the EvoApplications 2013, Lecture Notes in Computer Science (LNCS)*, volume 7835.

[7] S. Hutterer, F. Auinger, and M. Affenzeller. Evolutionary optimization of multi-agent control strategies for electric vehicle charging. In *Companion Publication of the 2012 Genetic and Evolutionary Computation Conference*.

[8] S. Hutterer, F. Auinger, M. Affenzeller, and G. Steinmaurer. Overview: A simulation based metaheuristic optimization approach to optimal power dispatch related to smart electric grids. In *Life System Modeling and Intelligent Computing (LNCS 6329)*, 2010.

[9] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, 1992.

[10] J. A. Momoh. Toward dynamic stochastic optimal power flow. In J. Si, A. Barto, W. Powell, and D. Wunsch, editors, *Handbook of Learning and Approximate Dynamic Programming*, pages 561–598. Wiley-Interscience, 2004.

[11] J. A. Momoh. *Electric Power System Applications of Optimization*. 2nd Edition, CRC / Taylor & Francis, 2009.

[12] M. Potter and K. De Jong. A cooperative coevolutionary approach to function optimization. In *Proceedings of the Third International Conference on Parallel Problem Solving from Nature (PPSN III)*. Springer-Verlag, 1994.

[13] M. Potter and K. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1), 2000.

[14] W. B. Powell, H. P. Simao, and B. Bouzaiene-Ayari. Approximate dynamic programming in transportation and logistics: A unified framework. *European Journal of Transportation and Logistics*, 1:237–284, 2012.

[15] C. Rosin and R. Belew. New methods for competitive coevolution. *Evolutionary Computation*, 5(1), 1996.

[16] H. Wang, C. E. Murillo-Sánchez, R. D. Zimmerman, and R. J. Thomas. On computational issues of market-based optimal power flow. *IEEE Transactions on Power Systems*, 22(3), 2007.

[17] P. J. Werbos. Adp: Goals, opportunities and principles. In J. Si, A. Barto, W. Powell, and D. Wunsch, editors, *Handbook of Learning and Approximate Dynamic Programming*, pages 3–44. Wiley-Interscience, 2004.

[18] P. J. Werbos. Computational intelligence for the smart grid - history, challenges, and opportunities. *IEEE Computational Intelligence Magazine*, 6(3), 2011.

[19] A. J. Wood and B. F. Wollenberg. *Power Generation, Operation, and Control, 2nd Edition*. Wiley-Interscience, 1996.

[20] R. D. Zimmerman, C. E. Murillo-Sanchez, and D. Gan. Matpower - a matlab power system simulation package; http://www.pserc.cornell.edu/matpower/#docs, Mar. 2013.