CHAPTER 4

PSO AND GA BASED PID CONTROLLER

This chapter emphasize the importance of tuning of PID controller for AVR system by Particle Swarm Optimization and Genetic Algorithm Methods. A brief review of simulation of tuning of PID controller by PSO and GA algorithm pertaining to the present work is also presented.

4.1 INTRODUCTION

Bio-inspired meta-heuristic algorithms are important tools for solving problems computationally. All computation involves algorithms, and the efficiency of an algorithm largely determines its usefulness. PSO and Genetic Algorithm (GA) are the two important algorithms for solving problems. PSO was developed by Kennedy & Eberhart in 1995. PSO is a population based search algorithm based on the simulation of the social behaviour of birds, bees or a school of fishes. This algorithm originally intends to graphically simulate the graceful and unpredictable choreography of a bird folk. Each individual within the swarm is represented by a vector in multidimensional search space. This vector has also one assigned vector which determines the next movement of the particle and is called the velocity vector. The PSO algorithm also determines how to update the velocity of a particle. The PSO has been proved to be a powerful competitor in the field of optimization.

Genetic algorithms are among the most popular evolutionary algorithms in terms of the diversity of their applications. A vast majority of well-known optimization problems have been solved using genetic algorithms. In addition, genetic algorithms are population-based and many modern
evolutionary algorithms are directly based on genetic algorithms or have some strong similarities to them.

The GA was developed by John Holland and his collaborators in the 1960s and 1970s. It is a model or abstraction of biological evolution based on Charles Darwin's theory of natural selection. Holland was probably the first to use the crossover and recombination, mutation and selection in the study of adaptive and artificial systems. These genetic operators form the essential part of the genetic algorithm as a problem-solving strategy. Since then, many variants of genetic algorithms have been developed and applied to a wide range of optimization problems.

In this research, an efficient optimization algorithm is proposed using PSO and GA for tuning the optimal parameters of PID controllers used for AVR of power generating systems. The primary aim of the controller is to maintain the voltage at an optimal level under varying operating conditions. The transient response AVR is very important, because both the amplitude and time duration of the response must be within the prescribed limits. The performance of AVR system with PSO and GA tuned PID controllers is analysed for its validity of the proposed method. The proposed method has better flexibility and good dynamic response than the conventional PID controller, thereby providing improved performance with respect to overshoot, settling time and oscillations.

4.2 BIO-INSPIRED ALGORITHM FOR AVR

In general, an electric power network is a large and complex system which consists of synchronous generators, transformers, transmission lines, relays and switches etc. Various control objectives such as operating conditions, actions, and design decisions require solving one or more linear or non-linear optimization problems. Bio-inspired techniques are considered as a
useful optimistic technique for deriving the global optimization solution for complex problems. Since the loads are switched on and off, the AVR is susceptible to sudden changes to its voltage. Under these circumstances, keeping voltage within the allowable range is one of the important tasks of AVR control. The PID control system with plant indicating AVR and Bio-inspired optimization techniques based PID is shown in Figure 4.1.

![Figure 4.1 PID control system with bio-inspired optimization techniques](image)

The $K_p$, $K_i$ and $K_d$ are respectively the proportional, integral and derivative gains of the PID controller that are tuned by Bio-inspired optimization algorithms. In the proposed system, PSO and GA algorithms are used to optimize set of PID parameters in the system to achieve desired output $y_d$. The control output 'u' from Bio-inspired optimization technique - PID is based on the error signal 'e', which is the difference between actual output 'y' and the desired output $y_d$. The objective on the PSO and GA based optimization is to seek a set of PID parameters such that the feedback control system has a minimum performance index. A set of optimal PID parameters can yield good dynamic performance is considered as a useful and promising
technique for deriving the global optimum solution of complex functions. Hence, application of these algorithms yields improved performance characteristics in terms of settling time, rise time and maximum peak overshoot. Bio-inspired optimization techniques like PSO and GA are applied to tune the controller gains to ensure optimal performance at nominal operating conditions.

4.3 OVERVIEW OF PSO ALGORITHM

Optimization algorithms are another area that has been receiving increased attention in the past few years. An optimization algorithm is a numerical method or algorithms for finding the maxima or the minima of a function operating with certain constraints. Computational intelligence is a successor of artificial intelligence relying on evolutionary computation, which is a famous optimization technique.

Computational intelligence finds it fundamental application in the area of fitness function design, methods for parameter control, and techniques for multimodal optimization. PSO is a computational algorithm technique based on swarm intelligence. This method is motivated by the observation of social interaction and animal behaviours such as fish schooling and bird flocking.

It mimics the way they find food by the cooperation and competition among the entire population. A swarm consists of individuals called particle, each of which represents a different possible set of the unknown parameters to be optimized. In PSO system, particle flies around in a multi-dimensional search space adjusting its position according to its own experience and the experience of its neighbouring particle. The goal is to effectively search the solution space by swarming the particles towards the best fitting solution encountered in previous iterations with inertia of
encountering better solution through the course of process and eventually converging on a single minimum or maximum solution. The performance of each particle is a measured accounting to a pre-defined fitness function, which is reported to the process being solved.

PSO is initialized with a group of random particles and then searches for optima by updating the particles in each generation. In every iteration, each particle is updated by two "best" values. The first one is the best solution achieved so far called $p_{\text{best}}$. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called $g_{\text{best}}$. After finding the two best values, the particle updates its velocity and positions. The above mentioned overview of PSO is depicted as shown in Figure 4.2.

![Figure 4.2 Modification of search point by PSO](image)

### 4.4 PERFORMANCE INDICES

The design of control system is an attempt to meet a set of specification which defines the overall performance of the system in terms of certain measurable quantities. The most common performance criteria are
Integrated Absolute Error (IAE), Integrated of Time weight square Error (ITSE) and Integrated of Square Error (ISE) that can be evaluated analytically in frequency domain. Each criterion has its own advantages and disadvantages. The disadvantage of IAE and ISE criteria is that its minimization can result in a response with relatively small overshoot but long settling time, because the ISE performance criteria weights all errors equally independent of time. The set of good control parameters can yield a good step response that will result in performance criteria minimization in the time domain, this performance criterion is called Fitness Function $F(k)$ which is expressed as follows:

$$F(k) = (1 - e^{-\beta})(M_p + e_{ss}) + e^{-\beta}(t_s - t_r)$$  \hspace{1cm} (4.1)$$

where

$M_p$ = Peak overshoot

$t_s$ = Settling time

$t_r$ = Rising time

$\beta$ = Weighting factor

$e_{ss}$ = Steady state error

In order to evaluate the effectiveness of the proposed algorithm, the evaluation function is introduced as:

$$\text{Evaluation function } f = \frac{1}{F(k)}$$  \hspace{1cm} (4.2)$$
4.5 **PSO-PID CONTROLLER**

With the development of computational methods in the recent years, optimization techniques are often proposed to tune the optimal control parameters. Metaheuristics algorithm can be applied for tuning of PID controller gains to confirm optimal control performance at normal operating conditions.

In the conventional PID controller, the gains are arbitrarily selected and the results are verified for every set of arbitrary gain values. PSO algorithm finds the Proportional, Integral and Derivative gains of the PID controller and the values are passed to the PID controller of AVR as shown in Figure 4.3.

![Figure 4.3 Block diagram of PSO-PID controller](image)

4.6 **ALGORITHM FOR PSO**

The algorithm of PSO include following steps:

1. Initialize the swarm by assigning random position and velocity to each particle.
2. Evaluate the fitness function for each particle.

3. Compare the current fitness value with $p_{\text{best}}$ value of the particle.

4. If current fitness value is better than the previous best value ($p_{\text{best}}$), then set this as current $p_{\text{best}}$.

5. Now best evaluated value of $p_{\text{best}}$ is set as $g_{\text{best}}$ value.

6. Update the velocity and position of the particle according to Equations (4.6) and (4.7).

7. Repeat the Steps 2 to 6 until suitably good stopping criterion is met such as maximum number of iteration or best fitness value.

The $i^{th}$ particle in the swarm is represented as in the $d$-dimensional space.

$$X_i = (X_{i,1}, X_{i,2}, X_{i,3}, \ldots, X_{i,d})$$ \hspace{1cm} (4.3)

The best previous position of the $i^{th}$ particle is represented as

$$P_{\text{best}} = (P_{\text{best},1}, P_{\text{best},2}, P_{\text{best},3}, \ldots, P_{\text{best},d})$$ \hspace{1cm} (4.4)

The index of the best particle among the group is $G_{\text{best},d}$

Velocity of the $i^{th}$ particle is represented as

$$V_i = (V_{i,1}, V_{i,2}, V_{i,3}, \ldots, V_{i,d})$$ \hspace{1cm} (4.5)

The updated velocity and the distance from $P_{\text{best},d}$ to $G_{\text{best},d}$ is given as
\[
V_{i,m}^{(t+1)} = W * V_{i,m}^t + C_1 * \text{rand()} * (P_{\text{best},m} - X_{i,m}^t) \\
+ C_2 * \text{rand()} * (G_{\text{best},m} - X_{i,m}^t)
\]

(4.6)

and

\[
X_{(i,m)}^{(t+1)} = X_{(i,m)}^{(t)} + V_{i,m}^{(t+1)}
\]

(4.7)

For \(i = 1, 2, 3, \ldots \ldots n, \ m = 1,2,3, \ldots \ldots d\)

where

\[
\begin{align*}
n & = \text{Number of particle in the group} \\
d & = \text{dimension index} \\
t & = \text{point of iteration} \\
V_{i,m}^t & = \text{velocity of particle at iteration } i \\
W & = \text{Inertia weight factor} \\
C_1,C_2 & = \text{Acceleration Constant} \\
\text{rand} () & = \text{Random number between 0 and 1} \\
X_{i,d}^{(t)} & = \text{Current position of the particle ‘i’ at iteration} \\
P_{\text{best},i} & = \text{Best previous position of the } i^{th} \text{ particle} \\
G_{\text{best},i} & = \text{Best particle among all the particles in the swarming population}
\end{align*}
\]

The best values used for various parameters in PSO implementation are listed in Table 4.1 and flow chart for PSO-PID algorithm is shown in Figure 4.4.
Figure 4.4 PSO algorithm based PID controller

Start

Initialize the population and other algorithm parameters

Initialize the Kp, Ki, Kd values randomly

Calculate the Fitness Function

Calculate the Local Best and Global Best

Maximum Iteration Reached?

NO

Yes

Output Parameters Kp, Ki, Kd

Stop
### 4.7 RESULTS OF PSO-PID CONTROLLER

The proposed algorithm is implemented as simulation using MATLAB SIMULINK. The block diagram of PSO-PID controller for AVR system using SIMULINK is shown in Figure 4.5.

The transfer function of AVR system with PSO-PID method is shown in Equation (4.8) and step response of AVR system using PSO-PID method is shown in Figure 4.6. PID parameters and results obtained by PSO method is shown in Table 4.2.

\[
G(s) = \frac{1.154s^2 + 4.538s + 2.552}{0.002s^3 + 0.067s^4 + 0.615s^3 + 2.704s^2 + 5.5538s + 2.552} \tag{4.8}
\]

<table>
<thead>
<tr>
<th>Method/Parameters</th>
<th>PSO based PID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proportional Gain (K_p)</td>
<td>0.4548</td>
</tr>
<tr>
<td>Integral Gain (K_i)</td>
<td>0.2552</td>
</tr>
<tr>
<td>Derivative Gain (K_d)</td>
<td>0.1154</td>
</tr>
<tr>
<td>Peak Overshoot (M)</td>
<td>3.62</td>
</tr>
<tr>
<td>Settling time (t_s) (sec)</td>
<td>2.43</td>
</tr>
<tr>
<td>Rise time (t_r) (sec)</td>
<td>0.435</td>
</tr>
<tr>
<td>Evaluation function (f)</td>
<td>1.1123</td>
</tr>
</tbody>
</table>
Figure 4.5  Simulink model of PSO-PID controller

Figure 4.6  Step response of PSO-PID controller
Figure 4.7  Servo response of PSO–PID controller

Figure 4.8  Servo response of PSO-PID controller for various set point value
4.8 OVERVIEW OF GENETIC ALGORITHM

Genetic Algorithm (GA) was first introduced by John Holland in 1960. It is a heuristic optimization technique inspired by the mechanisms of natural selection. GA starts with an initial population containing a number of chromosomes where each one represents a solution of the problem in which its performance is assessed based on a fitness function.

GAs manipulate not just one possible solution to a problem but a collection of possible solutions. This is called as population. The possible solution in the population is called chromosomes. These chromosomes are the encoded representations of all the parameters of the solution. Each chromosome is compared to other chromosomes in the population and awarded fitness rating that indicates how successful this chromosome to the latter.
To encode better solutions, the GA will use genetic operators or evolution operators, such as crossover and mutation for the creation of new chromosomes from the existing ones in the population. This is achieved by either merging the existing ones in the population or by modifying an existing chromosome.

The selection mechanism for parent chromosomes takes the fitness of the parent into account. This will ensure that the better solution will have a higher chance to procreate and donate their beneficial characteristic to their offspring. A genetic algorithm is typically initialized with a random population consisting of 20-100 individuals.

This population which is also known as mating pool is usually represented by a real-valued number or a binary string called a chromosome.

The objective function gives each individual a corresponding number called its fitness. The fitness of each chromosome is assessed and a survival of the fittest policy is applied. In this proposed work, the magnitude of the error will be used to assess the fitness of each chromosome. There are three main stages of a genetic algorithm. They are reproduction, crossover and mutation.

4.8.1 Reproduction

During the reproduction phase the fitness value of each chromosome is evaluated. This value is used in the selection process to provide bias towards fitter individuals. An example of common selection taking is the Roulette wheel selection method. Each individual in the population is allocated a section of a roulette wheel, the size of the selection is proportional to the fitness of the individual.
4.8.2 Crossover

Once the selection process is completed, the crossover algorithm is initiated. The crossover operation swaps certain parts of the two selected strings in a bid to capture the good parts of old chromosome and create better new ones. Genetic operation manipulates the character of a chromosome directly. Using the assumption, that certain individual gene code, on average, produce fitter individuals. The crossover probability indicates, how often crossover is performed. The simplest crossover technique is the single point crossover.

4.8.3 Mutation

Mutation prevents the algorithm to be trapped in local minima and maintains diversity in the population. Commonly lower mutation rate should be chosen. Higher mutation rate may probably cause searching process which will change into random search.

4.9 GA-PID CONTROLLER

Since designing of PID controller for AVR system by conventional methods gives oscillatory response, hence the controlled parameters obtained by conventional methods does not give optimum value. So Genetic algorithm is used to select the optimum PID gains in this proposed work. The block diagram of GA based PID controller is shown in Figure 4.10. For Optimizing PID parameters, the fitness function used in this proposed work is same as Equation (4.1) given in Chapter 4 for the purpose of comparison.
Figure 4.10  Block diagram of GA-PID controller

The steps involved in the implementation of GA-PID controller are as follows: Generate an initial random population of individuals for a fixed size.

1. Generate the initial population.

2. Evaluate their fitness.

3. Select the fittest members of the population.

4. Reproduce using a Roulette wheel selection method.

5. Implement crossover operation on the reproduced chromosome.

6. Implement mutation operation with low probability.

7. Repeat Step 2 until a predefined convergence condition is met.

The parameters used in GA implementation are listed in Table 4.3, and flow chart for GA-PID algorithm is shown in Figure 4.11.
Table 4.3  GA parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population Size</td>
<td>15</td>
</tr>
<tr>
<td>Crossover rate</td>
<td>0.70</td>
</tr>
<tr>
<td>Mutation rate</td>
<td>0.05</td>
</tr>
<tr>
<td>Generation</td>
<td>100</td>
</tr>
<tr>
<td>Selection</td>
<td>Roulette wheel</td>
</tr>
</tbody>
</table>

Figure 4.11  GA based PID controller
4.9.1 Results of GA-PID Controller

The simulation of the proposed algorithm has been done using MATLAB Simulink. The block diagram of GA-PID controller for AVR system using SIMULINK is shown in Figure 4.12.

![Block diagram of GA-PID controller](image)

**Figure 4.12  Simulink model of GA-PID controller**

The transfer function of AVR system with GA-PID method is shown in Equation (4.8) and step response of AVR system using GA-PID method is shown in Figure 4.13. PID gain values and results obtained by GA method is shown in Table 4.4.

\[
G(s) = \frac{2.16s^2 + 6.93s + 9.45}{0.002s^5 + 0.067s^4 + 0.615s^3 + 3.71s^2 + 7.93s + 8.45} \tag{4.9}
\]
Table 4.4  Parameters and results obtained by GA based tuning method

<table>
<thead>
<tr>
<th>Method/Parameters</th>
<th>PSO based PID Controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proportional Gain $K_p$</td>
<td>0.693</td>
</tr>
<tr>
<td>Integral Gain $K_i$</td>
<td>0.845</td>
</tr>
<tr>
<td>Derivative Gain $K_d$</td>
<td>0.216</td>
</tr>
<tr>
<td>Peak Overshoot $M_p(%)$</td>
<td>18.4%</td>
</tr>
<tr>
<td>Settling time $t_s$(sec)</td>
<td>1.92</td>
</tr>
<tr>
<td>Rise time $t_r$(sec)</td>
<td>0.254</td>
</tr>
<tr>
<td>Evaluation function $f$</td>
<td>0.8889</td>
</tr>
</tbody>
</table>

Figure 4.13  Step response of GA-PID controller
Figure 4.14  Step response of GA-PID controller

Figure 4.15  Servo response of GA-PID controller for various set point value
SUMMARY

In this chapter the optimal tuning of PID parameters by PSO and GA method has been implemented as simulation in SIMULINK under MATLAB. The performance of the two proposed algorithm is validated in terms of evaluation function. The results obtained from simulation by PSO algorithm have better performance than GA in time domain specification.