Effects of Particle Swarm Optimization and Genetic Algorithm Control Parameters on Overcurrent Relay Selectivity and Speed

SETHEMBISO NONJABULO LANGAZANE AND AKSHAY KUMAR SAHA, (Senior Member, IEEE)
Discipline of Electrical, Electronics, and Computer Engineering, University of KwaZulu-Natal, Durban 4041, South Africa
Corresponding author: Akshay Kumar Saha (saha@ukzn.ac.za)

ABSTRACT
Distribution systems continue to grow and becoming more complex with increasing operational challenges such as protection miscoordination. Initially, conventional methods were favoured to solve overcurrent relay coordination problems; however, the implementation of these methods is time-consuming. Therefore, recent studies have adopted the utilisation of particle swarm optimization and genetic algorithms to solve overcurrent relay coordination problems and maximise system selectivity and operational speed. Particle swarm optimization and genetic algorithms are evolutionary algorithms that at times suffer from premature convergence due to poor selection of control parameters. Consequently, this paper presents a comprehensive sensitivity analysis to evaluate the effect of the discrete control parameters on particle swarm optimizer and genetic algorithms performance, alternatively on the behaviour of overcurrent relays. Optimization algorithms aim to minimise overcurrent relay time multiplier settings and accomplish optimal protection coordination. The findings indicate that particle swarm optimization is more sensitive to inertia weight and swarm size while the number of iterations has minimal effect. The results also depict that 30% crossover, 2% mutation, and smaller population size yield faster convergence rate and optimise fitness function, which improves genetic algorithms performance. Sensitivity analysis results are verified by comparing the performance of particle swarm optimization with the genetic algorithms which show the former parameter setting outperforms the latter. The relay operational speed is reduced by 15% for particle swarm optimization and system selectivity is maximised. The optimal protection coordination achieved using particle swarm optimization showed superiority of the algorithm, its ability to circumvent premature convergence, consistency, and efficiency.

INDEX TERMS
Control parameter, genetic algorithms, overcurrent relay, particle swarm optimization, power system protection, protection coordination, selectivity, speed.

ABBREVIATION
AC Alternating Current.
CTI Coordination Time Interval.
DAPSO Dynamic Adaptive Particle Swarm Optimization.
ES Evolutionary State.
GA Genetic Algorithms.
IDMT Inverse Definite Minimum Time.
MAPSO Modified Adaptive Particle Swarm Optimization.
MDE Modified Differential Evolution.
PSO Particle Swarm Optimization.
PSO-DE Particle Swarm Optimization with Differential Evolution.
PSO-LDIW Particle Swarm Optimization with Linearly Decreasing Inertia Weight.
PSO-RIW Particle Swarm Optimization with Random Inertia Weight.
PSO-CIW Particle Swarm Optimization with Chaotic Inertia Weight.
PSO-TVAC Particle Swarm Optimization with Time-Varying Acceleration Coefficient.
PSM Plug Setting Multiplier.
TMS Time Multiplier Setting.
I. INTRODUCTION
Due to rising emphasis on substation automation, SCADA, and monitoring control [1], operational speed and protection coordination form the most important aspect and are prime factors in any protection system [2], [3]. As the demand for electricity continues to rise, distribution systems are taking a strain and becoming more complex with increasing loads, voltages, and currents [4]. As a result, protection miscoordination may occur due to poor overcurrent relay settings [5], [6]. Moreover, operational challenges such as a greater percentage of power network equipment damage and customer service disruptions caused by breakdowns and faults in the distribution feeders as overhead power systems are subjected to either partial or permanent faults [6]. Although systems are designed to be as fault-free as possible, it is impractical to eliminate the fault occurrence completely [7]. However, system abnormalities must be catered to during the engineering design stage, commissioning, and maintenance to circumvent enormous damage and guarantee the protection of expensive equipment [8], [9]. Protection coordination is of paramount importance since the failure of protective devices to operate under faulty conditions can damage some essential parts due to fire that may result from massive-short circuits; consequently, the system loses synchronism of the machinery and equipment [10], [11]. This necessitates the need to optimize overcurrent relay operating time and maximize selectivity [12].

For many years, power systems engineers and researchers relied on conventional optimization techniques to perform relay coordination. The disadvantage of the methods is that solutions are based on iterative trial and error, and the process is laborious as well as time-consuming [6], [10]. Hence, authors in [13] and [14] researchers advocated the need for utilising evolutionary algorithms to mitigate setbacks presented by conventional optimization methods. Also, in [15] and [16], the importance of employing evolutionary algorithms was emphasised. Another study in [17] also highlighted the need to utilise metaheuristic algorithms to overcome drawbacks presented by conventional techniques. Genetic algorithms (GA) and Particle Swarm Optimizer (PSO) have transpired as efficient and effective algorithms for handling coordination problems. Nevertheless, setting evolutionary algorithms control parameters to attain optimum overcurrent relay settings is a long-standing issue [18]. In the literature, a comparative study evaluating PSO algorithm performance with GA algorithm demonstrates the latter fails to perform efficiently [19]; it suffers from a condition referred to as premature convergence that causes it to lose diversity which results in population degenerating to the local solution. In order to enhance convergence rate and overcome premature convergence of GAs, the idea of varying mutation and crossover probability has been adopted in [17], [20]. The importance of selecting mutation probability and crossover probability such that GA algorithm performance is more efficient has been documented in GA research work [15]–[19], [21]. Rojas et al. [20] conducted a simple GA sensitivity analysis where one control parameter is varied at a time to study and observe the system performance, and it was found that mutation probability, crossover-type, and population size have minimal influence on GA algorithm performance, while crossover probability has a significant effect [20]. In [21], Charbonneau urged that mutation probability must increase as population diversity decreases and lessen as population diversity increases [21]. Arenas et al. [22] and [23] conducted a sensitivity analysis of GA through the utilisation of a combinational crossover and mutation probabilities, it was found that the combinational approach is effective and leads to an optimum global solution [22], [23]. However, the best optimal solution is obtained when incorporating a high crossover rate with a low mutation probability. A review providing a dynamic approach for choosing crossover and mutation probabilities is presented in [15], another study [16] conducted a review on the application of genetic operators highlighting the pros and cons of GA control parameters, whereas in [17], a review on genetic operators was presented stipulating past, present, and future techniques of evaluating GA algorithm.

In contrast, a study comparing PSO algorithm performance with other metaheuristic techniques depicts that PSO algorithm manages to obtain global minima with fewer iterations [24]. However, the former presented limitations such as getting trapped in local minima, thus failing to obtain a global solution. Consequently, papers in [24] and [25] conducted research work to improve PSO algorithm performance. An earlier study [25] introduced the weight term in the velocity update equation to sharpen particles’ search ability by balancing local search and global search. Inertia weight and swarm size are heavily linked with premature convergence. Kennedy and Eberhart [26] classified velocity equation into three components to regulate the impact of particles’ previous velocity to current speed. In [27], the behaviour of PSO particles was studied, it was suggested that both cognitive and social parameters in the velocity equation must not surpass 4 [27]. A further study in [28] presented a theoretical analysis when the acceleration constants surpassed 4, the particles displayed high levels of oscillations. These approaches serve as a guide and aids in improving swarm search capability and circumventing premature convergence [27], [28]. Another study in [29] analysed PSO performance by means of design experiments, this approach decreases the number of simulations runs; however, it does not permit the study of each control parameter, hence not much was established [13], [29]. As a result, this paper aims to provide a simple sensitivity analysis approach for both GA and PSO algorithms, comprehensive review, and comparison of algorithms in terms of convergence and fitness function values. Table 1 below presents the summary of the literature survey highlighting the gaps in the existing research and the contributions of the proposed paper. The sensitivity analysis is conducted by means of varying one control parameter at a time while keeping others constant. This approach allows users to distinguish parameters responsible for poor
TABLE 1. Summary of literature survey.

| Algorithm                  | Existing tuning of parameters approach                                                                 | Proposed approach                                                                 |
|----------------------------|----------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| Genetic Algorithms         | Ref. [16] tuned crossover and mutation probabilities simultaneously. Similarly, authors in [22] and [23] adopted a combinational strategy that varies both crossover and mutation probabilities at the same time to observe algorithm behaviour. These approaches only focused on crossover and mutation probabilities which are not the only genetic operators that affect the performance of GA. Due to the variation of parameters simultaneously, it was difficult to determine poor performing parameters. | This paper proposes the analysis of population size, crossover, and mutation rates. Also, a parametric sensitivity analysis that changes one parameter at a time while keeping others constant is proposed. Analyses are done based on the speed of convergence and fitness function. |
| Particle Swarm Optimization| Authors in [25] suggested the use of larger inertia weight in the beginning, thereafter, slowly decreasing to a minimal value. Nonetheless, the time-varying based inertia weight variation may not lead to a global optimal solution. In [27] and [28], cognitive and social parameters were tuned to enhance the performance of the PSO algorithm; however, not much improvement was established. | As a result, this paper proposes the utilisation of self-adaptive control parameters such as evolutionary state-based inertia weight, acceleration coefficients, and repulsion-based velocity update to further improve PSO performance. |

The contributions of the paper are as follows:

a) The determination of poor performing control parameters based on the convergence speed and fitness function values which determines the robustness, efficiency, and superiority of the algorithm.

b) Analysis of overcurrent relay response based on operational speed and system selectivity to evaluate whether protection coordination is accomplished.

c) An algorithm that enhances original PSO performance by making the control parameters adaptive, adopting evolutionary state-based inertia weight, and utilising the repulsion-based position update technique is proposed.

An overview of protection philosophy is presented in Section II. A brief overview of the PSO algorithm and its control parameters is provided in section III. Section IV provides a basic review of the GA algorithm and parameters associated with the algorithm. Optimization problem setup and protection system under study is presented in section V. Results and discussions are provided in section VI. Lastly, conclusive remarks, as well as future recommendations, is presented in section VII.

II. OVERVIEW OF PROTECTION PHILOSOPHY

Reliability of electrical protection in distribution systems is of paramount interest to maintain continuous power supply to end-users [30]. Schweitzer et al. [31] evaluated power line redundancy and reliability, and analysis of protection scheme selectivity, sensitivity, speed is not covered in [31]. In one version [32], analysis tools are utilized to evaluate all protection scheme characteristics, new methods to improve protection quality are also presented [32]. Protection scheme functional characteristics must meet the strict requirement of modern distribution systems, which lack redundancy and operate near security limits [6]. Hence, it is important to conduct parametric analysis in such a manner that protection system quality is maximized. Studies have proven that protection system characteristic parameters are not independent as two of them are more likely to decrease when the other one increases [13], [31]. Another study in [33] stated that it is impossible to attain selectivity simultaneously in a system configuration with multiple equivalent sources due to the similarity of currents seen by overcurrent relays. This stands to be proven in this research paper.

In cases of system disturbances caused by unexpected load changes, faults, and malfunction of equipment, protective devices are required to react fast, be selective and reliable [3], [34]. If abnormal conditions occur in a network segment, a protective system is needed to clear the fault instantly without affecting the healthy section and promptly segregate the faulty segment. Protective system includes devices such as relays, circuit breakers, and other circuit interrupters to discriminate faulty equipment. Circuit breakers operate to discern abnormal sections of the system when prompted to function by the relay, which senses, localizes a defect, and issue a command to the breaker for removal of defective section [6], [35]. Protective relays continuously monitor electrical quantities and do not inhibit fault occurrence on the system. There are various benefits and purposes that protection system must fulfil, which include [6], [34], [36]:

- Minimisation of fault duration on distribution systems.
- Disconnection of defective transformers, lines, and other apparatus.
• Reduce service outages to the minimal segment of the system.
• Improving system performance, stability, reliability, and service continuity.
• Protection of customers’ apparatus.

A. OVERCURRENT PROTECTION

Excessive current levels in distribution systems are due to system abnormalities. These high current levels can be utilized to characterize the presence of defects and aid to trigger protective device operation accordingly, which differ in design specifications and system complexity [34], [36]. Amongst other devices, the most common overcurrent protection devices are moulded case circuit breakers, thermomagnetic switches, overcurrent relays, and fuses [6], [36]. The moulded case circuit breakers and thermomagnetic switches consist of elementary operating mechanisms and are predominantly utilized to protect low-voltage equipment [3], [36]; similarly, fuses are also used to protect low voltages lines and distribution transformers [14], [36]. Power systems are normally protected by overcurrent relays against excessive currents [36], as overcurrent protection is more economical and thus favoured on a distribution level compared to differential and distance protection systems [37].

At times, the main protection devices malfunction in a distribution system due to failure in the breaker tripping mechanism, insufficient tripping voltage, or defective protective relay; hence, backup protection is required to prevent severe damage to the system [6], [35]. Time overcurrent relays form a backbone of any protection strategy as they can be installed as primary or backup protection and consist of algorithms to monitor system voltages and current signals from voltage transformer and current transformer, respectively [38]. The received magnitude of current is used as an indicator of abnormalities and trips if the incoming current signal exceeds the pickup current. Usually, overcurrent relay determines operating time by means of standard inverse characteristic curve emanating from inverse definite minimum time (IDMT) class. According to IEC 60255–151:2009 [39], detection of the overcurrent signal exceeds the pickup current. Usually, overcurrent relay determines operating time by means of standard inverse characteristic curve emanating from inverse definite minimum time (IDMT) class. According to IEC 60255–151:2009 [39], a relay’s pickup current is used as an indicator of abnormalities and trips if the incoming current signal exceeds the pickup current. Usually, overcurrent relays form a backbone of any protection strategy as they can be installed as primary or backup protection and consist of algorithms to monitor system voltages and current signals from voltage transformer and current transformer, respectively [38]. The received magnitude of current is used as an indicator of abnormalities and trips if the incoming current signal is detected as exceeding the pickup current. Usually, overcurrent relay determ

where \( T_{op} \), the time of operation, \( TMS \) is the time multiplier setting, and \( PSM \) is the plug setting multiplier. \( T_{op} \) values provide necessary protection coordination and depend on maximum fault current, IDMT curve type, and the downstream relays’ operating time [39]. For distribution systems, \( TMS \) normally ranges between 0.01 < \( TMS < 1.0 \), pickup current taken as 1.2 to 2 times the full load current, in step of 0.05 [19], [39]. The \( PSM \) is the ratio of maximum fault current in the relay coil to pick up current [32]. To achieve proper protection coordination, recent studies have explored different optimization approaches such as evolutionary algorithms to ensure correct sequential operation of downstream and upstream relays [6], [13]. Appropriate sequential functioning of primary and backup relays is guided by coordination time interval (CTI) which is a predefined allowable time between primary and backup relays and is dependent on circuit breaker speed, relay type, and other parameters [32]. For microprocessor-based relay, CTI is set between 0.1 seconds to 0.2 seconds, while electromagnetic relays utilize the range between 0.3 seconds to 0.4 seconds [19], [32].

To safeguard protective system reliability, the backup protection scheme must not operate under normal circumstances unless primary protection malfunctions. Once the CTI is exceeded, the backup relay must operate within coordination constraints as formulated in the following equation [6], [19]:

\[
T_{backup} - T_{main} \geq CTI
\]

where \( T_{backup} \) the backup protection operating time, \( T_{main} \) the time of operation for main protection. In this paper, the standard IDMT relay is utilized, and CTI of 0.4 is considered.

III. PARTICLE SWARM OPTIMIZER AND ITS CONTROL PARAMETERS

In PSO, each particle fly through the hyperdimensional design space at a random velocity initially, and its current position in the i-th dimension is denoted by \( s_i^{(k)} \) where \( k \) the iteration number, and \( i \) the individual particle. Each particle memorises its best position and its own experience denoted by \( pbest_i^{(k)} \), and the overall algorithms’ experience is denoted by \( gbest^{(k)} \). At each iteration, the particle velocity \( v_i^{(k)} \) is altered with current velocity and position from the personal best solution and the global best solution. Consequently, the \( v_i^{(k)} \) and \( s_i^{(k)} \) changes according to the following equations [40]:

\[
v_i^{(k+1)} = v_i^{(k)} + c_1 rand_1^{(k)} (pbest_i^{(k)} - s_i^{(k)}) + c_2 rand_2^{(k)} (gbest^{(k)} - s_i^{(k)}) \quad i = 1 \text{ to } N
\]

\[
s_i^{(k+1)} = s_i^{(k)} + v_i^{(k+1)} \quad i = 1 \text{ to } N
\]

where \( N \) is the swarm size, and \( rand_1^{(k)} \) and \( rand_2^{(k)} \) are two randomly generated numbers every \( k \) iteration with a range between 0 and 1. Acceleration coefficients \( c_1 \) and \( c_2 \), also referred to as the cognitive and social parameters, respectively, are positive constants [41]. In [26], the particle velocity update equation is classified into three terms, namely, the first term represents particles’ momentum which includes the impacts of the previous velocity on current velocity, the second fragment is associated with the cognitive component which signifies the pull of particles’ velocity towards its own personal best (pbest) while the third part represents the global best (gbest) or social interaction between particles [26]. After the calculation of particles’ new position and velocity, \( pbest_i^{(k)} \) and \( gbest^{(k)} \) are updated with the
following equations:

\[
p_{best}^{(k)} = \begin{cases} 
  s_i^{(k)}, & \text{if } f(s_i^{(k)}) < f(p_{best_i}^{(k)}) \\
  p_{best_i}^{(k)}, & \text{if } f(s_i^{(k)}) \geq f(p_{best_i}^{(k)}) 
\end{cases}
\]

\[
g_{best}^{(k)} = \begin{cases} 
  s_i^{(k)}, & \text{if } f(s_i^{(k)}) < f(g_{best}^{(k)}) \\
  g_{best}^{(k)}, & \text{if } f(s_i^{(k)}) \geq f(g_{best}^{(k)}) 
\end{cases}
\]

where \( f \) is the fitness function of the algorithm. Typically, the particles’ velocity value is fixed to the range \([-v_{max}, v_{max}]\) to mitigate the likelihood particles flying out of the search space [40]. Setting higher value for \( v_{max} \) results in particles fly past optimum solution, whereas smaller value leads to particles not exploring sufficiently search space hence particle gets trapped in local optima solution [40], [41].

### A. INERTIA WEIGHT

Due to limitations presented by \( v_{max} \), Shi and Eberhart [25] proposed the addition of weight term on the velocity update equation to sharpen particles’ searching ability by stabilising local search and global search [25]. The inertia weight (\( w \)) is the scaling factor correlated with the iteration velocity during the last time step and aims to improve PSO algorithm convergence rate. According to the modification proposed in [25], inertia weight is incorporated into equation (3) as follows:

\[
v_i^{(k+1)} = wv_i^{(k)} + c_1 r_1(p_{best_i}^{(k)} - s_i^{(k)}) + c_2 r_2(g_{best}^{(k)} - s_i^{(k)}) \tag{7}
\]

A higher inertia weight value promotes exploration, whereas a lower value facilitates exploitation which increases the local search capability of the PSO algorithm. An earlier study conducted in [43] proved experimental that larger maximum number of iterations \( \text{iter}_{max} \) increases computational time and it was seen that the selected value has a direct effect on the probability of PSO algorithm locating global solution [43]. Moreover, poor choice of selecting the number of iterations may lead to premature convergence. Too little number of iterations decreases the likelihood of the algorithm attaining global optimum solution whereas bigger maximum number of iterations improves convergence rate at the cost of computational time [43], [44].

### D. SWARM SIZE

Normally, swarm size, \( N \), is selected based on dimensionality and optimization problem complexity. It plays an essential role in PSO algorithm performance and have an impact on population diversity as it regulates the number of particles in the hyperdimensional space [40], [41]. Reference [45] stated that swarm size chosen between 5 and 10 particles is a good estimation; however, the utilisation of swarm ranging between 10 to 50 particles is common in solving optimization problems [45]. When the larger population size is selected, particles tend to discover more search space and PSO algorithm performs effectively and efficiently but at the expense of computational time [40], [45].

### E. PSO CONSTRAINT HANDLING MECHANISM

To avoid premature convergence and computational time presented by reinitialization of particles’ initial position approach implemented in [46], ref. [47] proposed the application of a penalty on infeasible solutions which resulted in PSO avoiding premature convergence. Richardson et al. [48] introduced two terms in the penalty function i.e., the amount at which constraint was violated and the amount of constraint violations. According to this modification, PSO cost function is calculated as [48]:

\[
F_1(x) = \begin{cases} 
  f_i(x), & \text{if feasible solution} \\
  f_i(x) + \beta_1 \left( \sum_{i=1}^{d} h_i \right) + \beta_2 \left( \sum_{i=1}^{d} y_i \right), & \text{If infeasible solution}
\end{cases}
\]

where \( F_1(x) \) is the penalty function, \( f_i(x) \) is the original cost function, \( \beta_1 \) and \( \beta_2 \) are penalty factors, \( \sum_{i=1}^{d} y_i \) is the sum of the amount \( d \) violated constraints, and \( \sum_{i=1}^{d} h_i \) is the sum.
of $d$ violated constraints. The penalty factors $\beta_1$ and $\beta_2$ are both set at $10^3$. This strategy penalises infeasible solutions by keeping track of constraint violations [48].

IV. OVERVIEW OF GENETIC ALGORITHMS

Genetic algorithms search solution space of a function by using survival of the fittest strategy, as opposed to PSO algorithm that is inspired by social behaviour of animals and births [13], [28]. The GA solutions initialise randomly to generate a new population by means of genetic parameters [13], [15]. Roulette wheel selection method allocates selection probability to each chromosome based on its fitness function value [49]. The randomly generated numbers are compared to the cumulative probability to determine the selection probability to each chromosome based on its fitness function value [49]. The randomly generated numbers are compared to the cumulative probability to determine the selection of a new population [28], [49]. This technique has a drawback of converging prematurely to local optimal due to the dominance of individuals that constantly succeeds in the competition and are chosen as a parent. The probability $P_i(t + 1)$ for each chromosome $i$ is defined in (10), where $f_i(t)$ is the fitness of chromosome $i$, and $n$ denotes population size [48], [49].

$$P_i(t + 1) = \frac{f_i(t)}{\sum_{j=1}^{n} f_j(t)}$$

Due to limitations presented by the roulette wheel method on genetic algorithms, extensions such as ranking method, scaling technique, and tournament selection were introduced to allow minimisation and negativity [49], [50]. In the ranking-based selection approach, the probability for each chromosome $P_i$ is assigned based on the succession of individual solution $i$ when all solutions are mapped by the fitness function to allow minimization. Chromosomes with higher fitness values have a great probability of appearing in the next generation. A number generated randomly between zero and one constitutes the reproduction of a new population $n_{keep}$ of feasible solutions. The probability of individual $P_i$ is determined as follows [49], [50]:

$$P_i = \frac{n_{keep} - i + 1}{\sum_{i=1}^{n_{keep}} i}$$

A. CROSSOVER OR RECOMBINATION

Subsequently, the fitness comparable selection approach has been employed to produce fitness-biased reproduction of the preceding generation, the crossover and mutation probabilities come into play [49], [50]. Crossover takes two individuals from the reproduced population pairs and applies recombination. Simple or single-point recombination generates a number randomly $r$ from a constant distribution and creates two new individuals ($x'_i$ and $y'_i$) according to the following equations [46]:

$$x'_i = \begin{cases} x_i & \text{if } i < r \\ y_i & \text{otherwise} \end{cases}$$

$$y'_i = \begin{cases} y_i & \text{if } i < r \\ x_i & \text{otherwise} \end{cases}$$

Crossover introduces new locality for supplementary execution within the hyperplanes, which are not signified by either parent arrangement [46], [50]. Therefore, the likelihood of obtaining greater performing offspring is considerably increased. High crossover probability results in the introduction of new structures into the population rapidly, whereas extremely high crossover causes discarding of structures quickly before selection generates enhancements [44]. If crossover probability is too small, the search stagnates due to the low exploration rate [44].

B. MUTATION

Mutation introduces heterogeneity into the population by expanding the search area that the GA algorithm evaluates and preventing GA algorithm from converging too fast before exploring the entire search space [49], [50]. Increasing mutation probability results in algorithm searching outside the current region of variable space which may impair the population by distorting existing good solutions. As a result, lower mutation rate is recommended [51]. Uniform mutation randomly selects one variable $j$ and equates it into a uniform random number $U(a_i, b_i)$ where $a_i$ and $b_i$ are lower and upper bound, respectively [51].

$$x'_i = \begin{cases} U(a_i, b_i), & \text{if } i < j \\ x_i, & \text{otherwise} \end{cases}$$

C. POPULATION SIZE

The group of chromosomes known as population affects the performance of GA algorithms. It was stated in [50] that a smaller population size leads to poor performance of the algorithm due to insufficient sample size for hyperplane exploration. Larger population discourages premature convergence by allowing more particles to cover the search space; however, at the cost of computational efforts [50], [51]. According to [52], anywhere between 10~50 is a good selection, but in other work, anywhere between 25~250 yields effective and efficient solutions to optimization problems [53].

D. GA CONSTRAINT HANDLING APPROACH

Parsopoulos et al. [54] proposed the utilization of a penalty factor to account for the sum of violated constraints and this technique is referred to as a non-stationary multistage assignment penalty function mechanism [54]. In [55], a strategy for managing constraint violation was not employed; as a result, overcurrent relays’ operating time was minimised, but the relays were not selective [55]. Reference [56] presents an improved constraint handling approach that incorporates a term that examines the number of constraints violated and increases a fitness value by a factor to penalise infeasible solution [56]. The same strategy is adopted in this work, stationary penalty function, $p$, penalises infeasible solution. Too big penalty function value results in the algorithm not recovering after being penalised, hence the value must be within average value [57]. All constraints are converted into
inequality as illustrated:

\[ p - \varepsilon \leq 0 \]  \hspace{1cm} (15)

\[ p_1 = W_1 \sum (-\Delta r_{mb}) \]  \hspace{1cm} (16)

\[ p_2 = W_2 \sum (TMS - 1) \]  \hspace{1cm} (17)

\[ p_3 = W_3 \sum (0.05 - TMS) \]  \hspace{1cm} (18)

where \( \varepsilon \) is the small tolerance value, \( W_1 \) controls the weighting of miscoordination penalty, \( W_2 \) controls the weighing of the upper bound penalty, and \( W_3 \) controls the weighting of the lower bound penalty [57]. The equations (19) and (20) show the objective function with the penalty factor incorporated. The \( n_{ih} \) penalty function \( p_n \), is added to the \( n_{ih} \) constraint function \( h_n \) only if constraint violation occurs [52].

\[ J = J + \max (0, p_n) \]  \hspace{1cm} (19)

\[ p_n = \begin{cases} \sum_{i=1}^{3} h_n, & \text{if } h_n < 0 \\ 0, & \text{otherwise} \end{cases} \]  \hspace{1cm} (20)

V. OPTIMIZATION PROBLEM SETUP

In this problem, the objective is to minimise the time multiplier setting values to accomplish optimum protection coordination in the distribution system. To investigate the effect of PSO and GA control parameters on overcurrent relay selectivity and speed, a distribution network layout developed in [58] is modified and utilised in this study for sensitivity analysis, as displayed in Fig. 1.

For PSO, three control parameters were considered for sensitivity analysis, that is, swarm size, number of iterations, and inertial weight. In contrast, GA considered population size, crossover probability, and mutation rate to evaluate algorithm sensitivity. The sensitivity analysis of the performance of algorithms is executed using overcurrent relay operating time. The best optimal solution is stored at the end of the individual run and compared after 10 executions to differentiate between the best and worst solutions attained. Algorithms and sensitivity analysis are implemented using software Matlab/Simulink.

VI. RESULTS AND DISCUSSIONS

In order to gain insight into overcurrent relay behaviour while easing sensitivity analysis efforts, the evaluation technique for comparing algorithms is as follows:

a) The algorithm that succeeded in obtaining the best fitness value is preferred, whereas any algorithm that yields poor performance due to premature convergence is not further considered.

b) The algorithm that determines the best global solution with the fewest iterations is preferred over the other.

c) Secondary to the speed of convergence is the efficiency and robustness of the algorithm – this is characterised by the lower number of iterations and diversity maintenance.

With respect to overcurrent protection characteristics, algorithms’ control parameters that maintain selectivity and optimize the speed of operation are preferred. Whereas any control parameter that yields two infeasible solutions, the one with a better fitness function value is favoured. Matlab/Simulink is utilised for modelling, computation, and demonstration of analyses. Convergence curves are computed to demonstrate PSO and GA control parameters performance under various conditions.

A. PSO SENSITIVITY ANALYSIS

In this sensitivity analysis study, PSO algorithm control parameters are considered through constraining particles to feasible areas. Irrespective of optimization problem nature, some of the control parameters’ values and choices have a major influence on PSO algorithm efficiency, and other control parameters have minimal or no effect [50], [59]. As discussed in the aforementioned section, the basic PSO parameters are swarm size, velocity components, number of iterations, acceleration coefficients, velocity clamping, and inertia weight. To examine PSO performance, only swarm size, number of iterations, and inertia weight are considered in this work.

1) SWARM SIZE

Swarm size sensitivity analysis conducted employs a range between 10 and 500 particles, acceleration coefficient is set at 2, maximum velocity and inertia weight is set at 50 and 0.9, respectively. The optimization problem is formulated with
an iteration of 1000 and it is imperative to note that all other control parameters are kept constant throughout the simulation. The effects of swarm size (N) on PSO algorithm performance are depicted in Fig. 2. It is noticeable that the incremental change of swarm size causes PSO to perform more effectively and efficiently; Nevertheless, more computational time is required to achieve the global optimum solution. Another observation of interest in Fig. 2 is that population size at 250 and 500 displays similar sensitivity, whereas performance slightly diverges for smaller sizes, i.e., 10 and 100. Due to a highly constrained optimization problem, where particles change based on their experience and the history of the whole swarm, the obtained results were expected. Additionally, larger variation between the enhancement for changing individual position and global best position either resulted in convergence at local optimum in lieu of global optimum or unnecessary wandering by individuals. An increase in swarm size rises the probability of particles settling to global minima and surpassing a definite threshold of which results to equivalent performance. Although the smaller swarm succeeded in obtaining global minima, the positioning of particles with different sizes varies.

A research paper [40] focusing on this section with swarm size range between 20 and 160 particles reported that swarm sizes have minimum influence on PSO algorithm performance [36]. However, it was observable that the performance of smaller sizes slightly differs from larger swarm sizes. A further study in [45], detailed sensitivity analysis with swarm size set between 25 and 500 particles, the study outcome proposed that the selection of swarm size must be made based on the number of variables [45].

The effect of swarm size on the overcurrent relay selectivity is investigated and verified. The convergence curve demonstrated that population size at 250 and 500 converges to the fitness of values of 3.39 seconds and 2.97 seconds, respectively. Consequently, time multiplier parameters at 500 particles are slightly smaller than swarm size set at 250, which means the sum of the relay operating time (i.e., speed of operation) is optimized and the system is more selective with maximum sensitivity. When abnormalities occur, it is detected by both primary and backup protection. The primary protection operates first as it consists of a shorter operating time compared to backup protection. For smaller swarm sizes set at 10 and 100 particles, the function converges to fitness values of 5.06 seconds and 4.45 seconds, separately. The behaviour demonstrates that overcurrent relays are taking too long to operate at 10 particles which violates one of the protection principles to isolate faults speedily. This implies that protection coordination is not accomplished, some of the overcurrent relays exceed coordination time interval (i.e., CTI ≥ 0.4). With swarm size set at 100, satisfactory optimum protection coordination is achieved in the distribution system. Also, selectivity is achieved, with only faulty equipment isolated promptly from the system within stipulated CTI.
3) INERTIA WEIGHT

Xin et al.
[43] proved experimentally that inertia weight ranging from 0.4 – 0.9 yields excellent results with improved efficiency and performance of PSO [43]. The linearly decreasing approach allows particles to explore broader search space in beginning and neighbouring areas in subsequent stages with reduced speed. It offers a substantial probability of reaching an optimum solution quickly [59]. Authors in [40] undertook a comprehensive study and applied inertia weight ranging between 0.8 – 1.2, and it was found that a larger inertia value facilitates global search, whereas a smaller inertia value improves local search [40]. The ultimate goal of inertia weight is the reduction of velocities or iteration and sharpening the exploration and exploitation ability of particles. Based on the aforementioned research work, three different ranges of inertia weight (i.e., W1 = 0.0 – 1.0, W2 = 0.8 – 1.2, and W3 = 0.9 – 0.4) are implemented. Swarm size is set at 100 particles, and 1000 iterations are utilised. To circumvent the effect initial population, 10 simulation runs are taken, acceleration coefficient parameters are set at 2, minimum and maximum velocity are 0 and 50, respectively. Fig. 4 depicts PSO sensitivity to various inertia weight values. At W1 = 0.0 – 1.0, PSO converges prematurely due to the decreased search abilities and particles getting trapped in local minima.

B. GA SENSITIVITY ANALYSIS

Setting genetic algorithms control parameters to obtain optimal solutions is a long-standing problem. Normally, control parameters are selected by the user with no guideline of what values might yield a better solution. Hence, a sensitivity analysis approach and verification with an experimental system is performed to establish suitable values of parameters and study the conceivable consequence of genetic operators and their impact on the performance of overcurrent relays. This study is inclusive of a parametric analysis of genetic parameters such as crossover probability, population size, and mutation probability. The goal is to evaluate the behaviour and determine optimal parameters for GA, which optimizes the time multiplier settings of overcurrent relays.

1) CROSSOVER AND MUTATION PROBABILITIES

Rojas et al.
[20] claimed that a number of crossover points (one, two, or uniform) have minimal effect on the performance of GA, while crossover probability, mutation rate, and population size have a significant influence [20]. Authors in [22] employed a combination of crossover and mutation probability [20% mutation, 80% crossover] and [10% mutation, 90% crossover], and it was found that the combination of mutation and crossover probability yields the best outcomes; however, mutation rate must be set at narrow range to avoid premature convergence, and too high mutation facilitates random search [22]. A further study in [53] displayed the superiority of crossover values ranging from 0.3 – 0.9, it was proven experimentally that bigger crossover probability (0.9) causes important individuals with better fitness values to get lost in the search space. This sensitivity study employs a single-point crossover range from 0.3 – 0.9, mutation rate range between 0.01 – 0.3, population size set at 100, and the number of generations set at 1000.

From Table 2, it is clear that the fitness value increases in proportion with the crossover and mutation probability, as expected. During crossover rate incremental simulations, the mutation probability was kept constant at the value of 2%. The dynamic performance of the considered crossover rates is depicted in Fig. 5 and the considered crossover ranges are presented in Table 2. At C = 0.3, the algorithm converges to the fitness function value of 2.30 seconds and when the crossover rate is set at C = 0.6, the algorithm converges to 3.95 seconds. At C = 0.9, larger convergence rate is obtained with an increased fitness function value of 5.37 seconds. Increasing the crossover rate resulted in an increase in the fitness function and convergence rate, which means overcurrent relays took too long to operate and the coordination time interval is exceeded on some relays.
In the second case, sensitivity analysis is conducted with four different mutation probability range from 0.02 – 0.3 and a constant single-point crossover probability of 30%. As depicted in Fig. 6, the parameters substantially affect the fitness function similarly to crossover probability. The incremental changes in mutation rate increase the fitness function and help to circumvent local optima through the prevention of chromosomes from being too identical to one another. For mutation probability set at 30%, the algorithm converges to the fitness function value of 2.80 seconds, whereas when the mutation probability is set at 20%, the algorithm converges to the fitness function of 2.25 seconds. At $M = 0.02$, faster convergence rate is obtained with a fitness value of 1.66 seconds. The fitness function drops rapidly from the initial value of 18.19 seconds to 4 seconds, where it starts to settle. It settles between 2.50 seconds and 1.30 seconds for about 250 iterations. The algorithm preserves population diversity and converges progressively until it reaches a fitness value that is within 8% of the final value (1.53 seconds) in 430 iterations. Final fitness function value of 1.66 seconds is reached in 910 iterations. From these findings, it is clear that larger mutation results in a slow convergence rate and higher fitness value. Whereas smaller mutation probability improves GA algorithm performance. This fulfils the purpose of mutation in genetic algorithms, which is to preserve and introduce diversity. Overall, overcurrent relays speed of operation is more optimized at 2% mutation rate, which means the relays are more selective and speedily when required to operate.

2) POPULATION SIZE
Population size is another contention that influences the categorization performance of the GA algorithms. In 2007, Lobo et al. [52] undertook a performance study considering known control parameters with respect to evolutionary algorithms [52]. It was found that larger population size increments parallelism which helps in finding solutions for

---

**TABLE 2. Comparison of time multiplier settings.**

| Time Multiplier Setting | Crossover Rate | Mutation Rate |
|-------------------------|----------------|---------------|
|                         | 0.3            | 0.4           | 0.5          | 0.6          | 0.7          | 0.8          | 0.9          | 0.02         | 0.1          | 0.2          | 0.3          |
| TMS 1                   | 0.33           | 0.49          | 0.57         | 0.75         | 0.77         | 0.79         | 0.83         | 0.35         | 0.45         | 0.36         | 0.42         |
| TMS 2                   | 0.34           | 0.45          | 0.51         | 0.73         | 0.69         | 0.75         | 0.79         | 0.22         | 0.37         | 0.33         | 0.39         |
| TMS 3                   | 0.29           | 0.43          | 0.47         | 0.69         | 0.65         | 0.65         | 0.66         | 0.18         | 0.17         | 0.29         | 0.33         |
| TMS 4                   | 0.23           | 0.36          | 0.43         | 0.51         | 0.62         | 0.59         | 0.63         | 0.14         | 0.19         | 0.25         | 0.35         |
| TMS 5                   | 0.21           | 0.28          | 0.37         | 0.30         | 0.51         | 0.57         | 0.57         | 0.21         | 0.23         | 0.19         | 0.30         |
| TMS 6                   | 0.19           | 0.19          | 0.33         | 0.25         | 0.43         | 0.49         | 0.50         | 0.17         | 0.13         | 0.19         | 0.29         |
| TMS 7                   | 0.17           | 0.17          | 0.20         | 0.19         | 0.31         | 0.43         | 0.45         | 0.15         | 0.11         | 0.17         | 0.21         |
| TMS 8                   | 0.18           | 0.18          | 0.18         | 0.17         | 0.23         | 0.31         | 0.31         | 0.31         | 0.09         | 0.06         | 0.22         | 0.17         |
| TMS 9                   | 0.15           | 0.15          | 0.15         | 0.15         | 0.17         | 0.25         | 0.29         | 0.02         | 0.12         | 0.15         | 0.19         |
| TMS 10                  | 0.08           | 0.07          | 0.13         | 0.08         | 0.11         | 0.13         | 0.21         | 0.11         | 0.07         | 0.07         | 0.13         |
| TMS 11                  | 0.13           | 0.02          | 0.07         | 0.13         | 0.02         | 0.02         | 0.13         | 0.02         | 0.02         | 0.02         | 0.02         |
| $\Sigma TMS$            | 2.30           | 2.72          | 3.41         | 3.95         | 4.51         | 4.98         | 5.37         | 1.66         | 1.92         | 2.25         | 2.80         |
complex optimization problems; however, it requires more valuations per generation, leading to an unacceptably slow convergence rate. Bakirli et al. [53] employed a range from 25 to 250 and claimed that the more population size increases, the fitness values also increase, similarly more computational effort is required [53]. A sensitivity analysis is conducted with population size ranging between 10 – 500, number of generations is set at 1000, mutation rate of 0.01, and single-point crossover. As anticipated, the results depict that by increasing population size, GA performs robustly and efficiently at the expense of computation time, which agrees with Bakirli et al. [53]. In Fig. 7, it is noticeable that larger population size (N = 500) succeeded in converging to the global minima with the fewest iteration number and hence managed to perform more efficiently than smaller population size (N = 10). Also, incremental changes in population size influences both exploitation and exploration which influences GA outcome greatly. The fitness function at N = 500 is smaller compared to the population size set at N = 10 and N = 100, meaning the algorithm convergence rate is faster, as shown in Fig.7. The algorithm maintains diversity and converges steadily until it reaches a fitness value that is within 10% of the final value. Another observation of interest is the slow convergence rate at N = 10 with maximum fitness function, which signifies the occurrence of premature convergence. With respect to protection coordination, overcurrent relays managed to operate promptly when population size is set at 500 and at N = 10 relays took too long to operate with coordination time interval longer than the stipulated value. This results in protection miscoordination and loss of selectivity as well as system reliability. When the population size is set at 100, efficient performance is achieved with a properly optimized speed of operation and coordination time interval is within the desired range.

From the experimental results of GA sensitivity analysis, it can be seen that the genetic operators, that is, population size, crossover, and mutation have a direct effect on the performance of genetic algorithms. An increase in population size, mutation probability, and crossover rate resulted in a slow convergence rate and higher fitness function. Whereas smaller probabilities of mutation and crossover improved GA performance with faster convergence rate and optimized fitness function. Similarly, a reduced population size yields higher fitness function and prompt rate of convergence to facilitate exploration and exploitation.

C. COMPARISON OF CONVERGENCE PERFORMANCE

Sensitivity analysis results are substantiated by comparing the GA algorithm performance with the PSO algorithm. As seen above, larger swarm size and iterations numbers increase computational time; hence, a swarm of 100 particles and iterations of 1000 are used to optimize relay operating times. GA is configured with the population size of 100 and the maximum number of generations is set at 1000. Optimization method robustness is determined by algorithms’ ability to avoid premature convergence, as depicted in Fig. 8. Optimization performance can only be appreciated after a certain number of iterations. In this research work, the simulation is made of 1000 iterations which are enough to appreciate any improvement of the algorithm. It is noticeable that GA and PSO converge to the fitness value of 3.554 seconds and 3.175 seconds, respectively. These fitness values clearly show that PSO algorithm convergence speed is slightly faster than GA, that is, in Fig. 8, the red curve (represents PSO) is quicker to reach its optimal solution than the blue curve (represents GA). Furthermore, it is good to observe that GA curves are smoother because the curve has fewer changes during convergence.

It can be deduced from the results that GA can exploit search space much more efficiently at the beginning of the search. The fitness value drops rapidly from the initial value of 20.20 seconds to 6 seconds, where it starts to settle, as shown in Fig. 8. It settles between 5.20 and 3.80 seconds for about 68 iterations. The algorithm preserves diversity and progressively converges until it reaches a fitness value within 5% of the final value (3.38 seconds) in 285 iterations. Final fitness value of 3.554 seconds was reached in 975 iterations. The PSO algorithm begins at a fitness value of 18 seconds.
and reaches a fitness value of 4 seconds in 79 iterations. It reaches the fitness value within 5% of the final fitness value after 250 iterations and slowly converges until it reaches the final fitness value of 3.175 seconds in 971 iterations. This shows the robustness and efficiency of the proposed optimization technique. The empirical study [11] claimed that with regards to performance, that is, attaining global best optimization technique. The empirical study [11] claimed that with regards to performance, that is, attaining global best solution, PSO algorithm yields the best result with even 100 particles which agrees with this study. Another comparative study [29] conducted with respect to GA found that PSO produces better optimum solutions than GA, which agrees with this study [29].

Table 3 shows TMS parameters for optimization techniques. Interestingly the data obtained is between 0.01 \sim 1.0, of which are the stipulated TMS values. Due to the factors discussed above, PSO provides TMS parameters for overcurrent relays that are slightly smaller than GA parameters.

Although PSO managed to perform efficiently and effectively, further modifications can be performed to improve PSO performance such that the operational speed and TMS values are further reduced. Sensitivity analysis results revealed that PSO is sensitive and dependent on its initial settings; particularly, inertia weight has the most influence on its performance. Different ranges of inertia weight were utilised and considered for sensitivity analysis, where it was seen that some parameters were unsuccessful in traversing particles leading to premature convergence. A significant number of researchers advocated the necessity of using larger inertia weight in the beginning, thereafter slowly decreasing to minimal value [25], [40]. Nonetheless, the time-based inertia weight was proposed to provide balance between exploration and exploitation search by enforcing the algorithm to retain feasible solutions only.

Table 3. TMS and time of operations for both algorithms.

| Relay No. | GA TMS | PSO TMS | TMS | $t_{op}$ |
|-----------|--------|---------|-----|----------|
| 1         | 0.72   | 0.66    | 0.66| 1.337    |
| 2         | 0.69   | 0.57    | 0.57| 1.384    |
| 3         | 0.57   | 0.50    | 0.50| 0.886    |
| 4         | 0.45   | 0.46    | 0.46| 0.706    |
| 5         | 0.30   | 0.28    | 0.28| 0.525    |
| 6         | 0.22   | 0.21    | 0.21| 0.392    |
| 7         | 0.18   | 0.17    | 0.17| 0.302    |
| 8         | 0.15   | 0.13    | 0.13| 0.258    |
| 9         | 0.13   | 0.11    | 0.11| 0.212    |
| 10        | 0.09   | 0.07    | 0.07| 0.126    |
| 11        | 0.05   | 0.02    | 0.02| 0.043    |
| Sum       | 3.55   | 7.239   | 3.18| 6.169    |

The modified adaptive particle swarm optimization (MAPSO) aims to attain distinctive inertia weight and acceleration coefficient values. MAPSO is a self-adaptive technique that uses feedback parameters produced by the fitness function of the individual particle. In [65], a chaotic-based non-linear inertia weight was proposed to provide balance between exploration and exploitation by reducing or increasing the search step [65]. However, the algorithm presented issues such as poor convergence, instability, and lack of feasible solutions. Another study in [66] proposed the use of evolutionary state-based inertia weight to balance exploration and exploitation. An evolutionary state (ES) is the mechanism used to self-automate the algorithm based on the environment as follows [65]:

$$ES_i^k = \frac{f(pbest_i^k) - f(gbest^k)}{f(x_i^k)}$$  \hspace{1cm} (21)

where $ES_i^k$ is the evolutionary estate, $i$ the individual particle, $k$ the iteration number, $f(pbest_i^k)$ the personal best solution fitness function, $f(gbest^k)$ is the global best fitness solution across the whole swarm, and $f(x_i^k)$ the fitness value of each particle current feasible solution [66]. Higher $ES_i^k$ value indicates the most recent feasible solution of an individual which results in algorithm ineffectiveness [62]. Therefore, it is of paramount importance to introduce modifications to the algorithm such that the convergence speed is improved and circumvent premature convergence. An adaptive strategy that modifies cognitive and social parameters, as well as inertia weight through observing current position and modifying control parameters, is employed in this work. A modified adaptive particle swarm optimization (MAPSO) previously proposed in [63] and [64] is modified and altered to best suit the overcurrent coordination problem. The technique was also proposed and implemented in [65] and [66]. The algorithm introduces an evolutionary state as a novel scheme to adapt control parameters such that relay operating time is reduced and the limitations presented by the original PSO are addressed through keeping track of particles’ current position with respect to its global best solution and personal best solution [66]. The three contributions presented by MAPSO are described as follows:

a) MAPSO is a constraint handling mechanism that enhances original PSO performance by making the control parameters adaptive and ensuring particles move towards feasible regions only.

b) An evolutionary state-based inertia weight is proposed to balance exploration and exploitation search by enforcing the algorithm to retain feasible solutions only.

c) A repulsion-based position update technique, as well as velocity reinitialization with respect to clamping limit, is adopted to enhance global exploration and increase robustness.
particle is near its personal best solution \( p_{best}^k \) and the global best solution \( g_{best}^k \) at the far end, this occurs when \( f(x_k^i) = f(p_{best}^k) \) [65], [66]. Smaller value of \( ES^k_i \) means either the most recent feasible solution of the individual particle is at the far end for both personal best and global best solutions or the personal best solution is near the global best. When \( f(p_{best}^k) = f(g_{best}^k) \), evolutionary state \( ES^k_i \) become zero [67]. This strategy yields optimum global solution and improves convergence thus, it is adopted in this research work to self-adapt inertia weight and evaluate the fitness function of each particle.

1) MAPSO - INERTIA WEIGHT

Yang et al. [68] propose a PSO algorithm with dynamic adaptation (DAPSO) that consists of two feedback parameters namely, aggregation factor which compares all particle performance with the best performing particle in the current iteration, and speed factor which evaluates the particles’ personal best solution were utilised to adapt inertia weight \( w_k^i \). Accordingly, inertia weight \( w_k^i \) was adapted using the following (22): where \( h_k^i \) is the speed factor, \( s \) the aggregation factor, \( w_i \) the initial inertia weight, \( a \) and \( \beta \) are system parameters with a range of \([0,1]\). This approach suffers from explosive divergence resulting in particles leaving the feasible region and never return, thus the algorithm is unstable and inefficient [68].

\[
w_k^i = w_s - a \left( 1 - h_k^i \right) + \beta s \tag{22}
\]

Another study [69] proposed a self-regulating inertia weight that controls each particle by increasing inertia weight value for the best performing particle while decreasing for all other particles. This scheme transpires from an idea that the best performing particle contains higher fitness value in its direction, hence, accelerates fast, whereas other all particles should proceed with a linearly decreasing inertia weight strategy. The self-regulating formula is given in (23), where \( w_k^i \) the inertia weight for \( i \)-th particle in the \( k \)-th iteration, \( \eta \) is a constant to control acceleration rate, \( w_{max} \) and \( w_{min} \) are maximum and minimum inertia weight. Harrison [70] in 2018 demonstrated that a self-regulating inertia weight approach can only lead to convergence behaviour when a certain threshold is known and is problem dependent, thus suggesting the use of particles’ fitness values in adapting inertia weight [70].

\[
w_k^i = \begin{cases} w_{k-1}^i + \eta \Delta w, & \text{for the best particle} \\ w_{k-1}^i - \Delta w, & \text{for all other particles} \end{cases} \tag{23}
\]

\[
\Delta w = \frac{w_{max} - w_{min}}{k} \tag{24}
\]

\[
w_k^i = w_{min} + \left( w_{max} - w_{min} \right) \left( \frac{\sum_{i=1}^{N} S^k_i}{N} \right) \tag{25}
\]

Equation (25) depicts adapting strategy based on particle success. This approach was proposed in [71] to evaluate particles’ behaviour such that the particle that improves its fitness at \( k \) iteration succeeds, whereas failure in enhancing fitness results in local minima solution [71]. \( N \) refers to swarm size and \( S^k \) a constant that is set to 1 if particle succeeds and 0 if unsuccessful. An increase in success percentage increases the inertia weight and decrease with decreasing success percentage. Other researchers [72] used non-linear function of decreasing inertia weight similar to the scheme developed in [73], which does not require a known iteration number. It is a new technique for updating inertia weight such that particles that obtain better solutions are considered for more exploitation capability. The scheme demonstrated a substantial improvement in the performance with regards to convergence speed and efficiency compared to dynamic adaptive particle swarm optimization DAPSO [73]. Although the above-mentioned variants improve the performance of the original PSO, the models become more complex due to the introduction of new parameters. Also, some adaptive variants are designed to solve unconstrained problems and suffer from premature convergence. Therefore, this paper proposes a constraint handling mechanism that improves original PSO performance by making the algorithm control parameters adaptive while ensuring the model is simple. In the proposed method, the evolutionary state \( ES^k_i \) behaves like inertia weight \( w_k^i \) hence, the performance of inertia weight is considered equivalent to evolutionary state [66].

2) MAPSO – ACCELERATION COEFFICIENTS

The movement of particle per iteration is controlled by acceleration coefficients, that is, both cognitive \( c_1^k \) and social \( c_2^k \) parameters. Typically, \( c_1^k \) and \( c_2^k \) are set at a constant value of 2.0 for the original PSO algorithm [74]; however, experimental results depicted that the employment of alternative configuration may yield better performance. It was proven that assigning different acceleration coefficient values results in improved performance and faster convergence [75]. Carlisle and Dozier [76] claimed that choosing larger cognitive parameter \( c_1^k \) than a social parameter \( c_2^k \) may lead to superior performance but with constraint \( c_1^k + c_2^k \leq 4 \). It was suggested in [75] that both cognitive and social parameters can be set as linearly decreasing values, but no improvement in performance was reported. Ratnaweera et al. [77] implemented PSO with time-varying acceleration coefficients (PSO-TVAC) such that \( c_2^k \) increases linearly over time while \( c_1^k \) decreases. The strategy aims to improve convergence by attracting more particles towards the global best solution. In [66], acceleration coefficients are influenced by evolutionary state instead of being time-based as follows [66]:

\[
c_1^k = \begin{cases} \frac{(c_{max} - c_{min}) \times (iter_{max} - k)}{iter_{max}} + c_{min} & \text{if } 0 \leq ES^k_i \leq 0.5 \\ c_{max} - \frac{(c_{max} - c_{min}) \times (iter_{max} - k)}{iter_{max}} & \text{if } 0.5 \leq ES^k_i \leq 1.0 \end{cases} \tag{26}
\]
The expression reinitialises a single component since other parameters in the velocity vector might contain a good structure which would allow the particle to move towards the global best solution with a decreasing velocity in the APSO-VI algorithm was introduced earlier in [80] to adapt inertia weight in a conversant particle swarm such that exploitation and exploration are regulated [80]. Authors in [66] proposed the reinitialization of velocity with respect to velocity clamping limit as given in the following equation.

\[ v_{d}^{k+1} = \begin{cases} \text{rand} \times v_{\text{max}}, & \text{if } v_{d}^{k+1} = 0 \\
\text{rand} \times (-v_{\text{max}}), & \text{if } v_{d}^{k+1} = 0 \\
\text{rand} \times v_{\text{max}}, & \text{if } v_{d}^{k+1} > 0 \\
\text{rand} \times (-v_{\text{max}}), & \text{if } v_{d}^{k+1} < 0 \end{cases} \quad (28) \]

The expression reinitialises a single component since other parameters in the velocity vector might contain a good structure which would allow the particle to move towards the global best solution with a decreasing velocity in the APSO-VI algorithm was introduced earlier in [80] to adapt inertia weight in a conversant particle swarm such that exploitation and exploration are regulated [80]. Authors in [66] proposed the reinitialization of velocity with respect to velocity clamping limit as given in the following equation.

\[
\begin{align*}
0 \leq ES_i^k \leq 0.5 & \text{ is regarded as a low evolutionary state in}\end{align*}
\]

0 \leq ES_i^k \leq 0.5 is regarded as a low evolutionary state in which the particle explores more global search at the beginning and towards the end, local search is encouraged [66]. Larger evolutionary state 0.5 \leq ES_i^k \leq 1.0 promotes exploitation in the beginning by permitting particles to converge toward the swarm’s best solution and progressively, more global exploration is encouraged towards the end [66].

The two constants \( c_{\text{min}} \) and \( c_{\text{max}} \) are set at 0 and 2.0, respectively. The same approach is adopted in this work to allow self-adapting acceleration coefficients to feasible regions.

3) VELOCITY UPDATE AND REINITIALIZATION

Pasupuleti and Battiti [78] introduced gregarious particle swarm optimization (G-PSO) which does not take into consideration particles’ previous velocity for determining new velocity [78]. The G-PSO population moves toward the global best position and once a particle gets trapped close to the global best solution, that particle realises with a random velocity [78]. Consequently, the algorithm continues exploring the local search while the original PSO proceeds by circumventing them. In [79], an adaptive parameter setting of particle swarm optimization based on velocity information (APS-OVI) was proposed, the algorithm uses current velocities of the particle to adapt inertia weight with the goal of getting velocity near to the ideal velocity [79]. The idea of a decreasing velocity in the APSO-VI algorithm was introduced earlier in [80] to adapt inertia weight in a conversant particle swarm such that exploitation and exploration are regulated [80]. Authors in [66] proposed the reinitialization of velocity with respect to velocity clamping limit as given in the following equation.

\[
\begin{align*}
0 \leq ES_i^k \leq 0.5 & \text{ is regarded as a low evolutionary state in}\end{align*}
\]

0 \leq ES_i^k \leq 0.5 is regarded as a low evolutionary state in which the particle explores more global search at the beginning and towards the end, local search is encouraged [66]. Larger evolutionary state 0.5 \leq ES_i^k \leq 1.0 promotes exploitation in the beginning by permitting particles to converge toward the swarm’s best solution and progressively, more global exploration is encouraged towards the end [66].

The two constants \( c_{\text{min}} \) and \( c_{\text{max}} \) are set at 0 and 2.0, respectively. The same approach is adopted in this work to allow self-adapting acceleration coefficients to feasible regions.

The expression reinitialises a single component since other parameters in the velocity vector might contain a good structure which would allow the particle to move towards the global best solution with a decreasing velocity in the APSO-VI algorithm was introduced earlier in [80] to adapt inertia weight in a conversant particle swarm such that exploitation and exploration are regulated [80]. Authors in [66] proposed the reinitialization of velocity with respect to velocity clamping limit as given in the following equation.

\[
\begin{align*}
0 \leq ES_i^k \leq 0.5 & \text{ is regarded as a low evolutionary state in}\end{align*}
\]

0 \leq ES_i^k \leq 0.5 is regarded as a low evolutionary state in which the particle explores more global search at the beginning and towards the end, local search is encouraged [66]. Larger evolutionary state 0.5 \leq ES_i^k \leq 1.0 promotes exploitation in the beginning by permitting particles to converge toward the swarm’s best solution and progressively, more global exploration is encouraged towards the end [66].

The two constants \( c_{\text{min}} \) and \( c_{\text{max}} \) are set at 0 and 2.0, respectively. The same approach is adopted in this work to allow self-adapting acceleration coefficients to feasible regions.

The expression reinitialises a single component since other parameters in the velocity vector might contain a good structure which would allow the particle to move towards the global best solution with a decreasing velocity in the APSO-VI algorithm was introduced earlier in [80] to adapt inertia weight in a conversant particle swarm such that exploitation and exploration are regulated [80]. Authors in [66] proposed the reinitialization of velocity with respect to velocity clamping limit as given in the following equation.

\[
\begin{align*}
0 \leq ES_i^k \leq 0.5 & \text{ is regarded as a low evolutionary state in}\end{align*}
\]

0 \leq ES_i^k \leq 0.5 is regarded as a low evolutionary state in which the particle explores more global search at the beginning and towards the end, local search is encouraged [66]. Larger evolutionary state 0.5 \leq ES_i^k \leq 1.0 promotes exploitation in the beginning by permitting particles to converge toward the swarm’s best solution and progressively, more global exploration is encouraged towards the end [66].

The two constants \( c_{\text{min}} \) and \( c_{\text{max}} \) are set at 0 and 2.0, respectively. The same approach is adopted in this work to allow self-adapting acceleration coefficients to feasible regions.

The expression reinitialises a single component since other parameters in the velocity vector might contain a good structure which would allow the particle to move towards the global best solution with a decreasing velocity in the APSO-VI algorithm was introduced earlier in [80] to adapt inertia weight in a conversant particle swarm such that exploitation and exploration are regulated [80]. Authors in [66] proposed the reinitialization of velocity with respect to velocity clamping limit as given in the following equation.

\[
\begin{align*}
0 \leq ES_i^k \leq 0.5 & \text{ is regarded as a low evolutionary state in}\end{align*}
\]

0 \leq ES_i^k \leq 0.5 is regarded as a low evolutionary state in which the particle explores more global search at the beginning and towards the end, local search is encouraged [66]. Larger evolutionary state 0.5 \leq ES_i^k \leq 1.0 promotes exploitation in the beginning by permitting particles to converge toward the swarm’s best solution and progressively, more global exploration is encouraged towards the end [66].

The two constants \( c_{\text{min}} \) and \( c_{\text{max}} \) are set at 0 and 2.0, respectively. The same approach is adopted in this work to allow self-adapting acceleration coefficients to feasible regions.

The expression reinitialises a single component since other parameters in the velocity vector might contain a good structure which would allow the particle to move towards the global best solution with a decreasing velocity in the APSO-VI algorithm was introduced earlier in [80] to adapt inertia weight in a conversant particle swarm such that exploitation and exploration are regulated [80]. Authors in [66] proposed the reinitialization of velocity with respect to velocity clamping limit as given in the following equation.

\[
\begin{align*}
0 \leq ES_i^k \leq 0.5 & \text{ is regarded as a low evolutionary state in}\end{align*}
\]

0 \leq ES_i^k \leq 0.5 is regarded as a low evolutionary state in which the particle explores more global search at the beginning and towards the end, local search is encouraged [66]. Larger evolutionary state 0.5 \leq ES_i^k \leq 1.0 promotes exploitation in the beginning by permitting particles to converge toward the swarm’s best solution and progressively, more global exploration is encouraged towards the end [66].

The two constants \( c_{\text{min}} \) and \( c_{\text{max}} \) are set at 0 and 2.0, respectively. The same approach is adopted in this work to allow self-adapting acceleration coefficients to feasible regions.
clamping-limit sensitivity analysis demonstrated that as particles explore more search space, the ability of the particle to fly past optimum solution increases, as depicted in Fig. 10. The figure shows that as the velocity clamping limit increases, the likelihood of obtaining a more feasible solution rises, resulting in quick convergence and more efficient algorithm performance.

5) COMPARISON BETWEEN MAPSO AND OTHER VARIANTS

In an attempt to overcome premature convergence in the PSO algorithm, Reference [81] “introduced a novel hybrid algorithm (PSO-DE) which integrates PSO with differential evolution (DE) to solve constraints by adopting a set of feasibility rules [81].” The PSO-DE algorithm provides better performance compared to modified differential evolution (MDE) [82] and differential evolution (DE) [83] hence, PSO-DE algorithm is utilised in this work for comparison purposes. “The PSO algorithm with linearly decreasing inertia weight (PSO-LDIW) was proposed in [25], [40] to linearly decrease the weight over time.” It was observed that PSO-LDIW algorithm convergence rate is slow toward global solution due to reduced inertia weight, which results in difficulty leaving the local optimum [40]. Another variant that integrates PSO with random inertia weight (PSO-RIW) was implemented in [84] and chaotic inertia weight (PSO-CIW) was proposed in [85]. Different PSO variants range are presented in Table 4. Fig. 11 depicts convergence curves for MAPSO and other variants.

MAPSO managed to outperform the original PSO, PSO-LDIW, PSO-RIW, and PSO-DE, as seen in Fig. 11. Furthermore, “MAPSO algorithm managed to attain the global optimum solution in the fewest iterations compared to other variants; this indicates the algorithms’ ability to converge” fast while avoiding premature convergence. Although MAPSO and PSO-CIW algorithm allowed particles to explore broader space with greater momentum, MAPSO performs better due to navigating the search space by means of evolutionary state whereas PSO-CIW navigates with respect to chaotic mapping which leads to stagnation. Variants such as PSO-LDIW, PSO-RIW, and PSO-DE failed to converge into the best global solution and were getting trapped in local optima. Other studies [81] found PSO-DE algorithm effective in solving the overcurrent relay coordination problem which disagrees with this work, as can be seen in Table 5, the algorithm yielded longer operational speed, and some relays were not selective which violates protection scheme principles.

Similarly, in [86] PSO-LDIW was compared with PSO-CIW which indicates a great difference between the algorithms, the study claimed that PSO-LDIW performs efficiently and robustly, is more stable with better global search capability than PSO-CIW which is contrasting with the results presented in this work. From Table 5 it can be seen that

![Figure 9: The effect of acceleration coefficients with respect to convergence.](image1)

![Figure 10: Velocity-clamping limit convergence curves.](image2)

![Figure 11: Convergence curves of MAPSO and other variants.](image3)

| PSO variants | Inertia weight range |
|--------------|---------------------|
| PSO-LDIW [25], [40] | 0.4 – 0.9 |
| PSO-RIW [84] | 0.5 – 1.0 |
| PSO-CIW [85] | 0.0 – 1.0 |

![Table 4: Range of considered variants.](image4)
TABLE 5. The sum of relay operating time and TMS values for each variant.

| Algorithms | Sum of TMS | Sum of operating time (s) |
|------------|------------|---------------------------|
| PSO-LDIW   | 4.52       | 8.603                     |
| PSO-DE     | 4.49       | 8.554                     |
| PSO-RIW    | 3.03       | 7.135                     |
| PSO-CTW    | 2.87       | 6.830                     |
| PSO        | 3.18       | 6.169                     |
| MAPSO      | 2.35       | 4.331                     |

MAPSO generates better values of TMS and relay operating time which further proves the algorithms’ superiority as compared to the previously proposed optimization algorithms. The operating time is further reduced from 6.169 seconds to 4.331 seconds which signifies the effectiveness of the newly proposed algorithm. All overcurrent relays preserve selectivity and protection coordination is achieved in the distribution system. This means abnormalities are removed as soon as possible without affecting the healthy section.

VII. CONCLUSION

The objective of this paper was to study the effects of particle swarm optimization and genetic algorithms control parameters on overcurrent relay sensitivity and speed. Due to the drawbacks of conventional methods, particle swarm optimizer and genetic algorithms are commonly used to solve the overcurrent relay optimization problems in distribution systems. However, setting evolutionary algorithms control parameters to obtain optimal relay settings is a long-standing problem. The sensitivity study conducted in this paper found that evolutionary algorithm control parameters certainly influence the performance of overcurrent relays. The altering of one parameter at a time while keeping others constant was very useful in finding parameters responsible for poor protection selectivity and speed of operation. The experimental results show a reduction in computational efforts and improvement in PSO convergence. The comparison of PSO with GA depicts that particle swarm optimizer converges faster than the genetic algorithms. The PSO algorithm reduces the overall TMS to the value of 3.18 and time of operation to 6.169 seconds, whereas GA yields an overall TMS value of 3.55 and the operational time of 7.239 seconds. This shows the PSO algorithm managed to optimize overcurrent relay settings and accomplished optimal protection coordination in distribution systems. The proposed MAPSO algorithm further improved operational speed and system selectivity. The sum of the primary relays’ operating time was further minimised from 6.169 seconds (original PSO) to 4.331 seconds (MAPSO). The comparative study verified the efficiency and effectiveness of MAPSO in solving overcurrent coordination problems, it showed that MAPSO outperforms all other variants. MAPSO yields the best optimal overcurrent relay settings with operating time of 4.331 seconds. Thus, it can be concluded that successful evaluation of control parameters and optimization of overcurrent relay settings was achieved. Future research work will focus on the impact of cognitive and social parameters as well as velocity clamping limit on original PSO algorithm performance. Moreover, sensitivity analysis can be done by means of altering two or more control parameters as opposed to the proposed strategy of one at a time approach.

REFERENCES

[1] M. L. De Klerk and A. K. Saha, “A review of the methods used to model traffic flow in a substation communication network,” IEEE Access, vol. 10, pp. 204545–204549, 2020.
[2] V. C. Mathebula and A. K. Saha, “Mission critical safety functions in IEC-61850 based substation automation system—A reliability review,” Int. J. Eng. Res. Afs., vol. 48, pp. 149–161, May 2020.
[3] S. Mnukwa and A. K. Saha, “SCADA and substation automation systems for the port of Durban power supply upgrade,” in Proc. Int. SAUPEC/RoMecha/PRASA Conf., Cape Town, South Africa, Jan. 2020, pp. 1–5.
[4] V. C. Mathebula and A. K. Saha, “Responsiveness of multi-channel IEC-61850 substation communication network reliability performance to changes in repair factors,” IEEE Access, vol. 9, pp. 789–800, 2021.
[5] V. C. Mathebula and A. K. Saha, “Reliability and availability of multi-channel IEC-61850 substation communication networks for mission-critical applications,” Int. J. Eng. Res. Afs., vol. 51, pp. 199–216, Nov. 2020.
[6] N. I. Nkhasi, “Real-time modelling and simulation of distribution system protection with and without renewable distributed generation,” M.S. thesis, Dept. Elect., Electron., Comput. Eng., Univ. KwaZulu-Natal, Durban, South Africa, 2017.
[7] V. C. Mathebula and A. K. Saha, “Impact of imperfect repairs and diagnostic coverage on the reliability of multi-channel IEC-61850 substation communication network,” IEEE Access, vol. 9, pp. 2758–2769, 2021.
[8] V. C. Mathebula and A. K. Saha, “Multi-state IEC-61850 substation communication network based on Markov and linear dynamical systems,” Sustain. Energy, Grids Netw., vol. 26, pp. 1–10, Jun. 2021.
[9] V. C. Mathebula and A. K. Saha, “Impact of quality of repairs and common cause failures on the reliability performance of intra-bay IEC 61850 substation communication network architecture based on Markov and linear dynamical systems,” IEEE Access, vol. 9, pp. 112805–112820, 2021.
[10] M. H. Hussain, S. R. A. Rahim, and I. Musrin, “Optimal overcurrent relay coordination: A review,” Proc. Eng., vol. 53, no. 1, pp. 332–336, 2013.
[11] S. T. P. Srinivas and S. K. Swarup, “Application of improved invasive weed optimization technique for optimally setting directional overcurrent relays in power systems,” Appl. Soft Comput., vol. 79, pp. 1–13, Jun. 2019.
[12] V. C. Mathebula and A. K. Saha, “Sensitivity and elasticity of multi-channel IEC-61850 substation communication networks to imperfect repairs,” Sustain. Energy, Grids Netw., vol. 26, pp. 1–12, Jun. 2021, doi: 10.1016/j.segan.2021.100443.
[13] A. R. Haron, A. Mahamed, and H. Shareef, “Coordination of overcurrent, directional and differential relays for the protection of microgrid system,” in Proc. Int. Conf. Electr. Electr. Informat., vol. 11, 2013, pp. 366–373.
[14] F. B. Bottura, W. M. S. Bernardes, M. Oleskovicz, and E. N. Asada, “Setting directional overcurrent protection parameters using hybrid GA optimizer,” Electr. Power Syst. Res., vol. 143, pp. 400–408, Feb. 2017.
[15] A. Hassanat, K. Almohammadi, and E. Abunawas, “Choosing mutation and crossover ratios for genetic algorithms—A review with a new dynamic approach,” Information, vol. 10, no. 12, pp. 390–399, 2019.
[16] C. K. H. Lee, “A review of applications of genetic algorithms in operations management,” Eng. Appl. Artif. Intell., vol. 76, pp. 1–12, Nov. 2018.
[17] S. Katoch, S. S. Chauhan, and V. Kumar, “A review on genetic algorithm: Past, present, and future,” Multimedia Tools Appl., vol. 80, no. 5, pp. 8091–8126, 2021.
R. C. Eberhart, Y. Shi,
“Comparing inertia weights and constriction factors in particle swarm optimization,” in Proc. Congr. Evol. Comput., La Jolla, CA, USA, Jun. 2000, pp. 84–88.

J. Xing, G. Chen, and Y. Hai,
“A particle swarm optimizer with multi-stage linearly-decreasing inertia weight,” in Proc. Int. Joint Conf. Comput. Sci. Optim., vol. 1, Xian, China, Apr. 2009, pp. 505–508.

A. P. Engelbrecht,
Computational Intelligence: An Introduction, 2nd ed. Hoboken, NJ, USA: Wiley, 2007.

J. S. Arora, Introduction to Optimum Design, 4th ed. New York, NY, USA: Academic, 2017.

X. Hu and R. Eberhart,
“Solving constrained nonlinear optimization problems with particle swarm optimization,” in Proc. 6th World Multiconf. Syst. Cybern. Inform., Orlando, FL, USA, Jul. 2002, pp. 203–206.

Z. Michalewicz and M. Schoenauer,
“Evolutionary algorithms for constrained parameter optimization problems,” Evol. Comput., vol. 4, no. 1, pp. 1–32, 1996.

J. T. Richardson, M. R. Palmer, G. E. Liepins, and M. Hilliard, “Some guidelines for genetic algorithms with penalty functions,” in Proc. Int. Conf. Genet. Algorithm., San Mateo, CA, USA, Jun. 1989, pp. 41–49.

J. H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligent. Cambridge, MA, USA: MIT Press, 1992.

R. L. Haupt and S. E. Haupt, Practical Genetic Algorithms. Hoboken, NJ, USA: Wiley, 2004.

Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs (AI Series). New Delhi, India: McGraw-Hill, 2011.

F. G. Lobo, C. F. Lima, and Z. Michalewicz, Parameter Setting in Evolutionary Algorithms, vol. 54. Berlin, Germany: Springer, Mar. 2007, pp. 1–18.

G. Bakhti, D. Birant, and A. Kut, “An incremental genetic algorithm for classification and sensitivity analysis of its parameters,” Expert Syst. Appl., vol. 38, no. 3, pp. 2609–2620, Mar. 2011.

K. E. Parsopoulos and M. N. Vrahatis, “Unified particle swarm optimization for solving constrained engineering optimization problems,” in Proc. Int. Conf. Natural Comput. Berlin, Germany: Springer, Aug. 2005, pp. 582–591.

C. W. So, K. K. Li, and K. Y. Fung, “Application of genetic algorithm to overcurrent relay grading coordination,” in Proc. 4th Int. Conf. Adv. Power Syst. Control, Oper. Manage., Hong Kong, 1997, pp. 283–287.

C. W. So and K. K. Li, “Intelligent method for protection coordination,” in Proc. IEEE Int. Conf. Electr. Utility Dereg., Restruct. Power Technol., vol. 1, Apr. 2004, pp. 378–382.

C. A. Coello, “Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art,” Comput. Methods Appl. Mech. Eng., vol. 191, nos. 11–12, pp. 1245–1287, Jan. 2002.

S. Mnukwa, “Upgrading existing power supply at the port of Durban,” M.S. thesis, Univ. KwaZulu-Natal, Durban, South Africa, 2019.

R. Poli, J. Kennedy, and T. Blackwell, “Particle swarm optimization,” Swarm Intell., vol. 1, no. 1, pp. 33–57, Jun. 2007.

R. Poli, “Analysis of the publications on the applications of particle swarm optimisation,” J. Artif. Evol. Appl., vol. 2008, pp. 1–10, Feb. 2008.

M. Arunugam, M. Rao, and A. Chandramohan, “A new and improved version of particle swarm optimization algorithm with global–local best parameters,” Knowl. Inf. Syst., vol. 16, no. 3, pp. 324–350, 2008.

G. Ciuprina, D. Ioan, and I. Munteanu, “Use of intelligent-particle swarm optimization in electromagnetics,” IEEE Trans. Magn., vol. 38, no. 2, pp. 1037–1040, Mar. 2002.

M. Farzinfar, M. Jazari, and F. Razavi, “A new approach for optimal coordination of distance and directional overcurrent relays using multiple embedded crossover PSO,” Int. J. Electr. Power Energy Syst., vol. 61, pp. 620–628, Oct. 2014.

A. I. Atteya, A. M. E. Zonkoly, and H. A. Ashour, “Optimal relay coordination of an adaptive protection scheme using modified PSO algorithm,” in Proc. Int. Middle East Power Syst. Conf., Cairo, Egypt, Dec. 2017, pp. 689–694.

J. Liu, X.-W. Zhang, and L.-P. Fu, “A modified particle swarm optimization using adaptive strategy,” Expert Syst. Appl., vol. 152, Aug. 2020, Art. no. 113535.

M. Isiet and M. Gadala, “Self-adapting control parameters in particle swarm optimization,” Appl. Soft Comput., vol. 83, Oct. 2019, Art. no. 105653.

A. I. Atteya, E. I. Zonkoly, and H. A. Ashour, “Adaptive protection scheme for optimally coordinated relay setting using modified PSO algorithm,” J. Power Tech., vol. 97, no. 5, pp. 463–480, 2017.
