IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received September 14, 2020, accepted September 16, 2020, date of publication September 21, 2020, date of current version October 1, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3025673

Optimization of Day-Ahead Energy Storage System Scheduling in Microgrid Using Genetic Algorithm and Particle Swarm Optimization

AJAY RAGHAVAN, PAARTH MAAN, AND AJITHA K. B. SHENOY
Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104, India
Corresponding author: Ajitha K. B. Shenoy (ajith.shenoy@manipal.edu)

ABSTRACT We present a day-ahead scheduling strategy for an Energy Storage System (ESS) in a microgrid using two algorithms - Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). The scheduling strategy aims to minimize the cost paid by consumers in a microgrid subject to dynamic pricing. We define an objective function for the optimization problem, present its search space, and study its structural properties. We prove that the search space has a magnification of at least $50 \times (B_c - B_d + 1)$, where B_c and B_d are the maximum depths of charge and discharge in an hour (in percentage) of the ESS respectively. In a simulation involving load, energy generation, and grid price forecasts for three microgrids of different sizes, we obtain ESS schedules that provide average cost reductions of 11.31% (using GA) and 14.31% (using PSO) over the ESS schedule obtained using Net Power Based Algorithm.

INDEX TERMS Microgrid, energy storage system, dynamic pricing, scheduling strategy, optimization, genetic algorithm, particle swarm optimization.

I. INTRODUCTION

A power grid is an interconnected network that transmits electricity from producers to consumers. Traditional power grids that are capable only of uni-directional transmission of power, i.e., from producers to consumers, are rapidly transitioning into a two-way power flow system using modern smart technology [1]. A smart grid uses bi-directional power transmission and information to create an automated and distributed energy transmission network [2]. The evolution of the smart grid is expected to come through the plug-and-play integration of smart microgrids [3].

A microgrid can be thought of as a scaled-down version of a power grid [4]. The U.S. Department of Energy defines a microgrid as:

> A group of interconnected loads and distributed energy resources within clearly defined electrical boundaries that acts as a single controllable entity with respect to the grid [5].

Microgrids provide a feasible way to incorporate smart grid technology in a bottom-up approach since a complete overhaul of the traditional grid is arduous. Microgrids typically incorporate renewable energy generation sources such as solar panels and wind turbines. Consequently, microgrids also incorporate Energy Storage Systems (ESSs) to store this renewable energy.

ESSs are primarily used to capture energy produced at one time and use it at a later time. An important aspect of an ESS in a microgrid is that it degrades with time due to frequent charge/discharge cycles [6]. To maximize the life of an ESS, a microgrid could take the following precautions:

- **Maintain Optimal State of Charge:** Conventional energy storage norms dictate that Li-ion batteries are maximally functional between 10-20% and 80-90% State of Charge (SoC). Going above or below these thresholds could reduce the life cycle of the battery and may also hamper power output.
- **Limit Depth of Charge/Discharge:** The Depth of Charge/Discharge is a real value that refers to the amount of energy that is cycled in or out of the battery in a time interval, expressed as a percentage of the total capacity of the battery. Most batteries have a physical limit as to how much they can charge/discharge in a time interval. However, to maximize battery life, it's recommended to stay well below this value. For example, in one hour, the maximal depth of discharge of a battery may be equal

The associate editor coordinating the review of this manuscript and approving it for publication was Lin Zhang.

to 30%. Any discharge value above this could damage the battery.

Dynamic pricing of electricity is increasingly receiving more attention as a way to induce peak shaving and load leveling, by typically pricing electricity higher during periods of high demand, thereby incentivizing consumers to reduce their demand at that time and reshaping the load curve. Reducing the peak demand and flattening the demand profile benefits the producers by reducing overall plant and capital cost requirements [2]. One form of dynamic pricing is Time-of-Use (TOU) pricing, where the price of a unit of electricity is dependent on the time of the day.

Consumers will typically find innovative ways to take advantage of these variable tariffs [7]. One such way is using the ESS to store energy when the tariff is low, and using the stored energy when the tariff is high. This requires an optimal scheduling of the ESS, i.e., determining when the ESS charges or discharges. Another way to take advantage of dynamic pricing is to shift certain loads to a different time period within a day, usually to a time interval when the grid price is low. This demand response strategy is called load shifting and multiple papers have dealt this with this topic [8]–[10]. Like the former method, this is also a complex optimization problem.

This article will deal with the former problem, i.e., optimizing the ESS schedule in a microgrid with dynamic pricing to minimize the total cost paid by consumers. To be precise, we have applied Genetic Algorithm (GA) [11] and Particle Swarm Optimization (PSO) [12] to optimize a day-ahead ESS schedule in a microgrid that's connected to a traditional one-way-power-flow grid that imposes dynamic pricing on the microgrid. This optimization problem requires a day-ahead hourly forecast of the load, generated power, and grid price. Even though load and energy generation forecasting is a difficult task [13], multiple works have proposed various approaches to forecasting with reasonable accuracy [14]–[17].

The rest of this article is organized as follows. In Subsection I-A, we list the main contributions of this work. In Subsection I-B, we look at some pertinent related work which may help the reader better understand this article. In Subsections I-C and I-D, we provide brief introductions to GA and PSO in order to better understand the algorithm we propose in Section III. In Section II, we formally define the problem, present its search space, and study its structural properties. In Section III, we present the optimization algorithms based on GA and PSO. In Section IV, we describe the datasets used, the parameters chosen for the algorithms, and the results obtained. We finally conclude in Section V.

A. CONTRIBUTIONS

The main contributions of this article are as follows:

- We define an objective function to optimize a microgrid's day-ahead ESS schedule.
- We define the search space for the objective function.
- We study the search space's structural properties in depth and prove its magnification to be at least $50 \cdot (B_c - B_d + 1)$, where B_c and B_d are the maximum depths of charge and discharge of the ESS respectively.
- We design an algorithm to find an optimal or near-optimal solution to the objective function using Genetic Algorithm and Particle Swarm Optimization.
- We obtain an average cost reduction of 14.3% over the Net Power Based Algorithm, thus potentially providing an incentive to shift to smart grids and incorporate renewable energy infrastructure.

B. RELATED WORK

In [18], Youn and Cho optimize the operation of an Energy Storage Unit in a small power producing facility to minimize the total energy purchase from the power gird under spot prices. They make use of a conventional linear programming technique. Similarly, in [19], Maly and Kwan optimize the charge schedule of an ESS using Dynamic Programming. In [20], Mallol-Poyato *et al.* employ a two-stage evolutionary algorithm to first optimize a microgrid's structural parameters - the size of the ESS, the power ratings of the wind turbines and photovoltaic cells, etc., and then optimize the scheduling of the ESS considering a variable electricity pricing scenario. In [21], C.D. Korkas *et al.* present a scalable algorithm to manage demand response (adjusting the power consumption of a consumer to better match the demand for power on the supplier) and guarantee thermal comfort (by controlling HVAC systems) in microgrids. In [22], the authors use Gravitational Search Algorithm to develop a complete Energy Management System (EMS) for a microgrid and demonstrate its effectiveness over a conventional EMS. In [23], the authors propose a bi-level optimized scheduling strategy (day-ahead) for the decision making of a Distributed System Operator (DSO) and reconfigurable multi-microgrids to reduce the total system cost. They also investigate the proposed model on a real-test system under varying conditions. More papers related to microgrid scheduling and demand response management can be found in [24]–[28].

C. GENETIC ALGORITHM

Genetic Algorithm (GA) is a meta-heuristic typically used to solve non-linear optimization problems. It is based on Charles Darwin's Theory of Evolution through Natural Selection.

In GA, we begin with a set of candidate solutions for an objective function. These candidate solutions are called chromosomes. These chromosomes represent the first "generation".

A cost function then evaluates how fit each chromosome is. If the problem is a minimization problem, then a lower value of the cost function denotes higher fitness and vice versa. We then allow a fixed number of fittest chromosomes to "reproduce" with each other and produce offspring using a defined crossover operator. During the crossover, we also emulate mutation as it happens in real life, i.e, a defect in copying genes from parent to offspring. The mutation

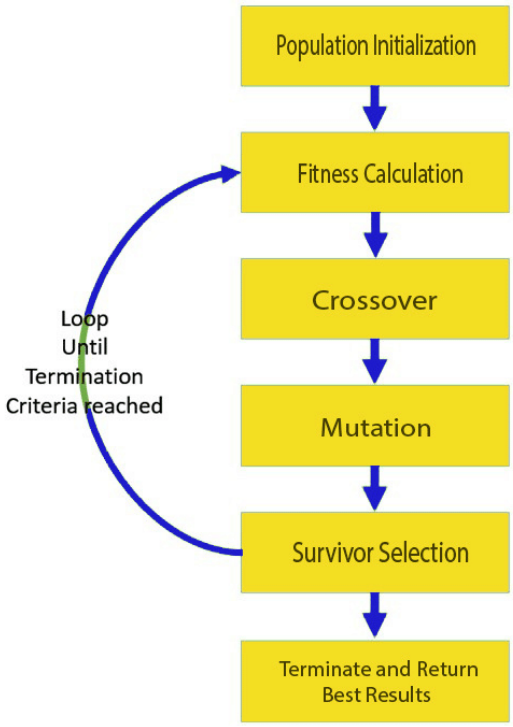

FIGURE 1. GA flowchart [29].

operator is hoped to reduce the probability that a solution remains stuck in local optima. We then add the offspring to the population.

The fittest chromosomes among the resulting population then continue to the next generation. This is repeated for 100s or 1000s of generations until an optimal solution is found or a termination criterion is met. Figure 1 depicts the steps in GA as a flowchart.

D. PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization (PSO) is a meta-heuristic that was first proposed by James Kennedy and Russell Eberhart in 1995 to solve non-linear optimization problems [12].

In PSO, we begin with a randomized set of candidate solutions, dubbed a "swarm" of particles. Each particle can be visualized as a point in a D-dimensional space, where D is the dimension of candidate solutions. Each particle then moves through the search space (thus exploring different possible solutions) according to simple mathematical formulae involving its position and velocity. A particle's movement is influenced by three parts - its "momentum", its "memory" of the best position it has been at (the position where the particle had the highest fitness value, called the local best), and the position of the fittest particle (called the global best). This is expected to guide the swarm toward optimal solutions. Note that a particle's fitness value is determined by a problem-specific fitness function.

Mathematically, say particle i is represented as $X_i = (x_{i1}, x_{i2}, .., x_{iD})$. Particle i's memory of the "best" position it traversed is represented as $P_i = (p_{i1}, p_{i2}, ..p_{iD})$. Let the rate at which the particle i moves (velocity) be given by $V_i = (v_{i1}, v_{i2}, ..v_{iD})$. Let the index of the particle with the best global position be g. Then, in each iteration of the algorithm, particles are manipulated according to Equations 1 and 2.

$$v_{id} = w * v_{id} + c_1 * r_1 * (p_{id} - x_{id}) + c_2 * r_2 * (p_{gd} - x_{id}), \tag{1}$$

$$x_{id} = x_{id} + v_{id} \tag{2}$$

In Equation 1, $w * v_{id}$ represents the momentum of a particle. Here, w is called the inertia factor. If the inertia factor is too high, the algorithm will have trouble converging on the optimal solution due to high momentum causing particles to move excessively in their own direction. On the other hand, if w is too low, the algorithm may prematurely settle on a non-optimal solution because the particles do not have enough momentum to sufficiently explore the search space. The inertia factor can also be dynamic (such as linearly decreasing with each iteration), and [30] has shown that it may lead to better results. In a linearly decreasing model, the inertia factor w varies in each iteration according to Equation 3.

$$w = w_{min} + \left(\frac{max_{iter} - iter}{max_{iter}}\right) \cdot (w_{max} - w_{min}) \tag{3}$$

In Equation 3, $iter$ represents the current iteration number, max_{iter} represents the total number of iterations, and w_{min} and w_{max} are the minimum and maximum values of the inertia factor respectively.

In Equation 1, $c_1 * r_1 * (p_{id} - x_{id})$ and $c_2 * r_2 * (p_{gd} - x_{id})$ represent the "cognitive" and "social" parts of the equation respectively. The cognitive part makes a particle move towards the "local best", and the social part makes a particle move towards the "global best". c_1 and c_2 are two positive constants known as acceleration coefficients, which control how the particles move toward the local best and global best positions respectively. If $c_1 > c_2$, the particle gravitates more towards the local best. If $c_2 > c_1$, the particle gravitates more towards the global best. r_1 and r_2 are two real numbers randomly generated in the range [0,1]. In each iteration of the algorithm, all particles move through the search space using Equations 1 and 2. We stop when we obtain a sufficiently good solution, or when there is no further improvement.

Meta-heuristics like PSO and GA are domain-independent randomized search heuristics which can be applied to any optimization problem. Practitioners usually use these heuristics when polynomial-time algorithms do not exist for an optimization problem. Surprisingly, for many of these problems, meta-heuristics return great solutions, even though solution optimality is not guaranteed. In the worst case, meta-heuristic algorithms may end up searching the entire search space (like brute-force). It should also be noted that since meta-heuristics like GA and PSO are highly probabilistic in nature, it is very hard to perform run-time analysis on them. There's only a small amount of literature available regarding the run-time analysis of simpler versions of GA [31]–[34] or PSO [35], [36], which are based on well known basic optimization problems like One Max Problem (maximizing the number of ones

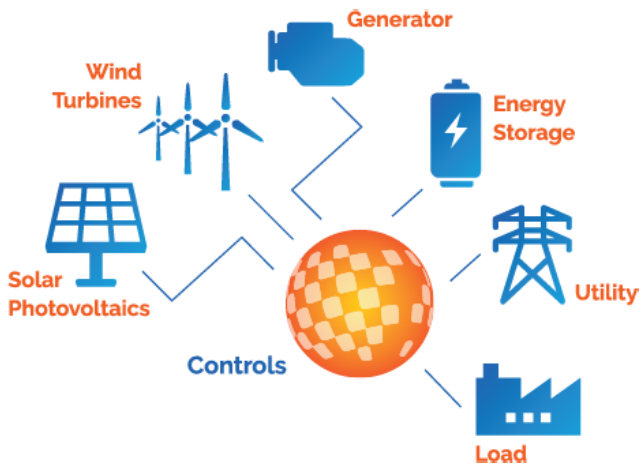

FIGURE 2. Generic micro-grid model [45].

in an n bit string). There is no literature regarding the run-time analysis of GA or PSO for hard non-linear optimization problems so far. For these hard optimization problems, practitioners apply different randomized search heuristics like GA, PSO, Ant Colony Optimization, and other bio-inspired algorithms and select the one that returns the best results for the specific problem. In the electrical engineering and energy domain, GA [37]–[41] and PSO [42]–[44] have been applied to many optimization problems and the results are encouraging. This is why we have compared the performance of GA and PSO for optimizing a day-ahead ESS schedule in a microgrid.

II. PROBLEM DEFINITION AND SEARCH SPACE

In this section, we formally define the problem and its search space. We also study the structural properties of the search space.

Consider a micro-grid μ (see Figure 2) comprising a variable number of solar panels, wind turbines, and (optionally) other forms of energy generators. The micro-grid also consists of a centralized ESS of capacity C (in kWh). The micro-grid is plugged into a traditional one-way-power-flow grid (called Utility in Figure 2). This traditional grid (will be called "main grid" from here on) subjects the micro-grid to hourly dynamic pricing of electricity. The load in Figure 2 is the sum total of loads of all the homes in the micro-grid. We optimize the charge/discharge scheduling of the ESS for a single day divided into hourly intervals such that the cost paid by the consumers in the micro-grid is minimized.

Mathematically, let L_i represent the cumulative load demanded by the microgrid (in kWh), G_i represent the cumulative energy generated by the renewable sources in the microgrid (in kWh), and A_i represent the unit price for electricity borrowed from the main grid (in cents/kWh), during the hour interval i, where $1 <= i <= 24$.

Let the maximum depth of charge of the ESS in one interval be B_c. Similarly, let the maximum depth of discharge be B_d. Then, we define a battery schedule vector as follows:

Definition 1 (Battery Schedule Vector): A battery schedule vector is a 24-dimensional real-valued vector $[S_1, S_2, \ldots, S_{24}]$ that represents a day-ahead schedule of a battery in a microgrid. Each S_i represents the depth of charge/discharge in hour interval i and satisfies the following constraints:

$$S_i \in \begin{cases} [max(-1 * (L_i - G_i)/C * 100, B_d), B_c] & L_i > G_i \\ [0, B_c] & G_i >= L_i \end{cases} \tag{4}$$

$$10 \leq \sum_{i=1}^{k} S_i \leq 90 \;\; \forall \;\; k \in [1, 24] \tag{5}$$

The constraint in Equation 4 ensures that the battery doesn't discharge more than $|B_d|$ percent and charge more than $|B_c|$ percent in an hour interval. The values of B_d and B_c are dependent on the battery used. It also ensures that the battery doesn't discharge in an interval where the generated energy is greater than the load demanded. The constraint in Equation 5 ensures that the battery's SoC doesn't dip below 10% or go above 90%, to maintain an optimal state of charge.

To obtain the cost associated with a battery schedule vector, we first define the electricity consumption vector.

Definition 2 (Electricity Consumption Vector): An electricity consumption vector is a 24-dimensional real-valued vector $[E_1, E_2, \ldots, E_{24}]$, where each E_i represents the amount of electricity borrowed by the microgrid from the main grid (in kWh). E_i is given by:

$$E_i = max(L_i - G_i + 0.01 * C * S_i, 0) \tag{6}$$

In Equation 6, $0.01 * C * S_i$ represents the total inflow or outflow of energy from the battery (depending on the sign of S_i) during the interval i. L_i - G_i represents either the remaining energy required to fully power the load (when $L_i > G_i$,) or represents the excess energy remaining (possibly to charge the battery) after G_i is used to power the load (when $L_i < G_i$). The 'max' ensures that E_i doesn't become negative (when $G_i > L_i$ and the battery cannot fully store the excess energy due to constraints specified in Equation 4 and Equation 5) since we assume that power only flows into the grid, not out of it.

Then, the optimization problem is to minimize the total cost paid by the consumer, P, which is the sum of the product of E_i and the unit electricity price during interval i, A_i, across all intervals.

$$MinimizeP = \sum_{i=1}^{24} E_i * A_i \tag{7}$$

Now we define the problem's search space. Note that we assume that the value of depth of charge or discharge increments or decrements in steps of 0.01. For example, in our problem, the SoC of the ESS can increment from 30% to 30.01%, but no real-value in between. Batteries in the real world also have similar minimum step values for changes in their SoC. We call this minimum depth of charge/discharge.

Definition 3 (Search Space): The search space Ω is defined as:

- *Search Space Elements: The search space is a set of all 24-dimensional vectors, say* $U = [S_1, S_2, \ldots, S_{24}]$, *where each* S_i *is a real number truncated to two decimal places in the range* $[B_d, B_c]$, *where* B_d *and* B_c *represent the maximum depth of discharge and charge in an hour interval respectively.*
- *Neighborhood structure: Two elements in the search space are neighbors to each other if they differ in one coordinate.*
- *Cost: For any search space element, say* $U = [S_1, S_2, \ldots, S_{24}]$, *the cost of U, denoted as* $Cost(U)$, *is defined as*

$$Cost(U) = \begin{cases} \sum_{i=1}^{24} E_i * A_i & \text{If } U \text{ satisfies constraints in Equations 4 and 5} \\ \infty & \text{Otherwise} \end{cases} \tag{8}$$

A_i denotes the grid price during the interval i and E_i is calculated using Equation 6.

Once again, note that we have defined the search space elements as real numbers truncated to two decimal places in the range $[B_d, B_c]$. This ensures that our search space has a finite number of discrete elements. Otherwise, our search space would consist of an infinite number of elements. Our search space has $\left[100 \cdot (B_c - B_d + 1)\right]^{24}$ elements. The constraints in Equation 4 and 5 are enforced by the way we have defined the cost function for the search space elements (an invalid element would have ∞ cost). Now we look at the structural properties of the defined search space:

Proposition 4: Each element in the search space Ω *has* $2400(B_c - B_d + 1)$ *number of neighbors.*

Proof: Let $U = [S_1, S_2, \ldots, S_{24}] \in \Omega$ and $V = [S'_1, S'_2, \ldots, S'_{24}] \in \Omega$ be neighbours. Then as per the Definition 3, U and V differ in one coordinate. Each S_i is a real value truncated to two decimal places in the interval $[B_d, B_c]$. This implies that each S_i can be one of $100(B_c - B_d + 1)$ values. There are 24 such S_is. Therefore, U has $2400(B_c - B_d + 1)$ neighbours. □

In the next proposition, we show that there is a $O(1)$-length path between any two elements in the search space.

Lemma 5: There is a $O(1)$-length path between any two elements in the search space.

Proof: Let $U = [S_1, S_2, \ldots, S_{24}]$ and $V = [S'_1, S'_2, \ldots, S'_{24}]$ be two search space elements. Let $i_1 < i_2 < \ldots < i_{r-1} < i_r$ indicate the indices where U and V differ. Then $U = U_{i_0} \rightarrow U_{i_1} \rightarrow \ldots \rightarrow U_{i_r} = V$ denotes the path from U to V, where $U_{i_{(j+1)}}$ is obtained from U_{i_j} by replacing the value of $S_{i_{(j+1)}}$ with $S'_{i_{(j+1)}}$. Hence the proposition. □

From Proposition 5, it is clear that the diameter of the search space or search graph is bound by $O(1)$. Next, we prove an important structural property of the search space called magnification. Magnification defines the number of outgoing edges from a considered cut-set. More magnification implies more number of edges going out from a cut-set, which would allow any heuristic algorithm to profitably make use of the defined search space. By the word profitably, we convey a high chance of a large ergodic flow going out from a cut-set in the search space. Now we formally define the term *magnification*:

Definition 6 (Magnification [46], [47, Definition II.10]): Let S be a non-empty subset of the search space Ω *and* $|S| \leq \frac{|\Omega|}{2}$. *Then, the magnification* $\mu(\Omega)$ *of the search space* Ω *is defined as:* $\mu(\Omega) = \min_S\{\mu(S)\}$, *where*

$$\mu(S) = \frac{|E(S, \overline{S})|}{|S|}$$

$E(S, \overline{S})$ *denotes the set of edges that go out from S to its complement* $\overline{S}$.

To find the lower bound on magnification, we make use of the canonical path method [46], [47]. In the canonical path method, we first need to define a unique path (say $\Gamma_{U,V}$) between any two search space elements U and V. Then we find an upper bound on the number of canonical paths that pass through an edge in the search space. Using that upper bound, we find the lower bound of the magnification. In the following theorem, we find the lower bound of the magnification of the proposed search space using the concept of canonical paths.

Theorem 7: The magnification of the proposed search space is at least $50 \cdot (B_c - B_d + 1)$.

Proof: Let $U = [S_1, \ldots, S_{24}]$ and $V = [S'_1, \ldots, S'_{24}]$ be any two distinct elements in the search space. The canonical path between U and V is defined in Lemma 5. Let R and S be two neighbours in the search space which differ at a unique index j. Consider any canonical path from U to V (say $\Gamma_{U,V}$) which makes use of the edge (R, S). By definition of canonical path, this is only possible if

$$R = [S'_1, S'_2, \ldots, S'_{j-1}, S_j, S_{j+1}, \ldots, S_{24}]$$

and

$$S = S'_1, S'_2, \ldots, S'_j, S_{j+1}, \ldots, S_{24}].$$

That is, the first j coordinates are identical in V and S, and the last $24-j+1$ coordinates are identical in U and R. Therefore, the total number of canonical paths passing through the edge (R, S) is $\left[100(B_c - B_d + 1)\right]^{(24-j)} \cdot \left[100(B_c - B_d + 1)\right]^{(j-1)} = \left[100(B_c - B_d + 1)\right]^{23}$. Now we prove the lower bound on magnification.

Let S be a non-empty subset of the search space Ω and $|S| \leq \frac{|\Omega|}{2}$. Let $\Phi_{(S,\overline{S})}$ denote a set of canonical paths $\Gamma_{U,V}$, which start at some vertex $U \in S$ and end at some vertex $V \in \overline{S}$. Then,

$$|\Phi_{(S,\overline{S})}| = |S| \times |\overline{S}| \geq \frac{|S| \cdot |\Omega|}{2} \text{ (since } |S| \leq \frac{|\Omega|}{2}) \tag{9}$$

Now consider an edge $(R, S) \in E(S, \overline{S})$. Every path in $|S| \leq \frac{|\Omega|}{2}$ should pass through one such edge from $E(S, \overline{S})$. We have shown that maximum number of canonical paths passing through an edge is $\left[100(B_c - B_d + 1)\right]^{23}$. Therefore,

$$|E(S, \overline{S})| \cdot \left[100(B_c - B_d + 1)\right]^{23} \geq |\Phi_{(S,\overline{S})}| \tag{10}$$

Using Equation 9 and Equation 10, we get

$$\frac{|E(S,\overline{S})|}{|S|} \geq \frac{|\Omega|}{2\big[100(B_c - B_d + 1)\big]^{23}}$$

Since $|\Omega| = \big[100(B_c - B_d + 1)\big]^{24}$, we get $\mu(S) = \frac{|E(S,\overline{S})|}{|S|} \geq 50 \cdot (B_c - B_d + 1)$. Hence the theorem. □

III. GA AND PSO FOR OPTIMIZING A DAY-AHEAD ESS SCHEDULE

In this section, we describe GA and PSO for the problem defined in Section II, using the search space defined in Definition 3. The first subsection describes GA and the second subsection describes PSO.

A. GA

Let n_p denote the population size. Initially, we randomly select n_p individuals from search space. In each generation, we select the fittest n_s individuals from the population to produce offspring. The fitness of a search space element is calculated as defined in Equation 8. Since the optimization problem is a minimization problem, one individual is fitter than another if it has a lower value of the cost function. Using n_s selected individuals, we can form $C(n_s, 2)$ pairs. Crossover is applied to each pair of individuals with probability P_c (according to Definition 8), resulting in n_c total offspring. After crossover, mutation is applied to the offspring produced with mutation probability P_m (according to Definition 9). Then, out of the total $n_p + n_c$ chromosomes, the fittest n_p proceed to the next generation. This process repeats for n_g generations. After the last generation, the fittest individual is returned. The crossover and mutation functions are defined as follows:

Definition 8 (Crossover): Let $U = [S_1, \ldots, S_{24}]$ and $V = [S'_1, \ldots, S'_{24}]$ denote parents. Then, the crossover operation will produce two children, C_1 and C_2, where

$$C_1 = [S_1, \ldots, S_{12}, S'_{13}, \ldots, S'_{24}]$$

and

$$C_2 = [S'_1, \ldots, S'_{12}, S_{13}, \ldots, S_{24}]$$

Definition 9 (Mutation): Let $U = [S_1, \ldots, S_{24}]$ denote a search space element. Mutation on U is defined as replacing any one of the dimensions, say S_i, with a real value (truncated to 2 decimal places) in the range provided by equation 4.

The complete Genetic Algorithm can be found in Algorithm 1.

Algorithm 1 GA for Optimizing Day-Ahead ESS Schedule

```
population = randomly select n_p elements from the search space (as defined in Definition 3)
mutation probability = P_m, crossover probability = P_c
generations = n_g and iteration = 0
while (iteration < generations) do
    Pick the fittest n_s individuals from population
    for each pair p1,p2 from C(n_s, 2) pairs do
        Apply crossover with probability P_c (Definition 8) and generate offspring
        Mutate the offspring with probability P_m (Definition 9)
    end for
    Out of the total resulting individuals (n_p + n_c), pick n_p fittest individuals to proceed to the next generation (n_c denotes the number of offspring generated in the iteration)
    iteration = iteration + 1
end while
return the fittest individual in population
```

B. PSO

A search space element as defined in Definition 3 is called a particle in PSO. We begin with a "population" array of m_p particles by randomly selecting elements from the search space. Every particle's initial velocity is set to $v_{initial}$. Each particle contains two additional attributes - localBest, which is a search space element denoting the best position (highest fitness value) the particle has traversed hitherto, and globalBest, the index of the particle in the population array with the highest fitness value. Let the number of iterations be max_{iter}. In each iteration, we move the particle through the search space using Equations 1 and 2. We pick different values of inertia factor w, including a linearly decreasing model, and obtain different results (will be discussed in Section IV). The values chosen for the acceleration coefficients c_1 and c_2 will also be revealed in Section IV. Algorithm 2 describes the whole process.

IV. RESULTS AND DISCUSSION

We obtained the hourly load (L), generated energy(G), and grid price (A) profiles over a 24-hour interval from [48]. In this data-set, there are microgrids of three different sizes - 8 hmes, 20 homes, and 40 homes - having batteries of cumulative sizes 43.44kWh, 108.6kWh, and 217.2kWh respectively. In this simulation, we assume that the cumulative batteries form a centralized ESS in each of the microgrids. We also assume that the SoC of each ESS is 30% at the start of the day. The value of B_d, the maximum depth of discharge, is -23 (which means SoC can fall by at most 23% in an hour interval). Similarly, B_c, the maximum depth of charge, is $+23$. Therefore, the search space contains $\big(100(B_c - B_d + 1)\big)^{24} = 4700^{24} \approx 2^{288}$ number of elements. Figure 3 shows how the forecast data vary for Home 1 in the microgrid with eight homes.

Before we look at the results using GA and PSO, we require a standard for comparison. For this, we use the Net Power Based Algorithm (NPBA), where the ESS charges or discharges to accommodate the difference between the load and the generated energy without regard to the electricity price. For example, in some cases, it may be prudent to save battery power for later when the grid price is higher, instead of using it right away. The total cost obtained for each of the three microgrids in the data-set using NPBA is shown in Table 1.

Algorithm 2 PSO for Optimizing a Day-Ahead ESS Schedule

population = randomly initialize m_p particles (search space elements)
cycles = max_{iter}
for each particle in population **do**
    Set velocity as $v_{initial}$
    Set localBest to particle's current position
end for
Set globalBest as index of particle with highest fitness value
if linearly decreasing inertia model **then**
    Set w_{min}
    Set w_{max}
else
    Set w
end if
Set acceleration coefficients c_1 and c_2
for *iter* in 1..cycles **do**
    if linearly decreasing inertia model **then**
        $w = w_{min} + ((max_{iter} - iter)/max_{iter}) * (w_{max} - w_{min})$
    end if
    for each particle in population **do**
        for each dimension in particle **do**
            Set r_1 and r_2 as randomly chosen real numbers in range [0,1]
            Set velocity = w * velocity + c_1 * r_1 * (localBest - current position) + c_2 * r_2 * (position of globalBest particle - current position)
            Set current position = current position + velocity
        end for
        if particle satisfies constraints in Equation 4 and 5 **then**
            Calculate fitness value of particle as defined in Equation 7
            if current position fitness is higher than localBest fitness **then**
                Set localBest to current position
                if current position fitness is higher than globalBest fitness **then**
                Set globalBest to index of current particle
                end if
            end if
        else
            Revert particle to previous position, since new position is invalid
        end if
    end for
end for
return the fittest particle in population

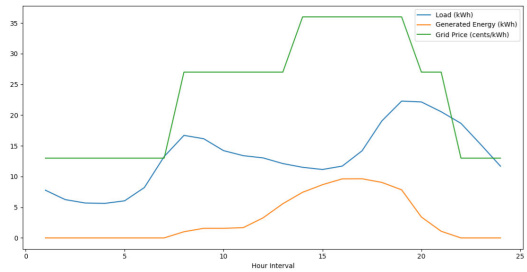

FIGURE 3. Forecast data for Home 1.

TABLE 1. Cost obtained using NPBA.

Microgrid Size (Homes)	NPBA Cost (Cents)
8	5471.22
20	13678.06
40	27356.12

TABLE 2. Cost obtained using GA with $n_p = 1000$, $n_s = 32$, $n_g = 1000$.

Run	Cost (Cents)
1	4804.72
2	4842.23
3	4922.91
4	4861.54
5	4834.86
6	4835.72
7	4882.54
8	4957.86
9	4854.10
10	4856.18
Average	4865.27

Now, let us first look at the results obtained using GA and PSO for the 8-home microgrid.

A. RESULTS USING GENETIC ALGORITHM

Parameters have a large impact on the performance of meta-heuristics. Therefore, we experimented with a wide range of parameter values and used trial and error to converge on the values that provided us with the best cost reduction.

We ran Algorithm 1 with different values of n_p, n_s, n_g, P_m, and P_c. We experimented with different values of P_m, the mutation probability, in the range [0, 0.2], and obtained best results with $P_m = 0.1$. Similarly, we found the best results with the crossover probability, P_c, set to 1. Tables 2, 3, and 4 depict the results for different values of n_p, n_s, and n_g. It can be seen that we obtained the best results with the population size, n_p, set to 1000, the number of parents picked to produce offspring in each generation, n_s, set to 32, and the number of generations, n_g, set to 1000. Since GA is a stochastic algorithm, we obtained different cost values in each run. The average cost obtained, 4865.27 cents (depicted in Table 2), is an improvement of 11.07% over NPBA. In most cases, GA virtually stagnated after 10-20 generations and entirely stagnated after about 900 generations. Figure 4 depicts an average run of GA. Figure 5 depicts how

TABLE 3. Cost obtained using GA with $n_p = 500$, $n_s = 23$, $n_g = 1950$.

Run	Cost (Cents)
1	4849.37
2	4895.22
3	4893.54
4	5029.34
5	4856.96
6	4788.81
7	4949.07
8	4871.18
9	4838.75
10	4900.28
Average	4887.25

TABLE 4. Cost obtained using GA with $n_p = 5$, $n_s = 2$, and $n_g = 500000$.

Run	Cost (Cents)
1	5353.70
2	5571.86
3	5038.64
4	5206.28
5	5332.69
6	4934.82
7	5114.27
8	5277.62
9	5112.73
10	5461.7
Average	5240.44

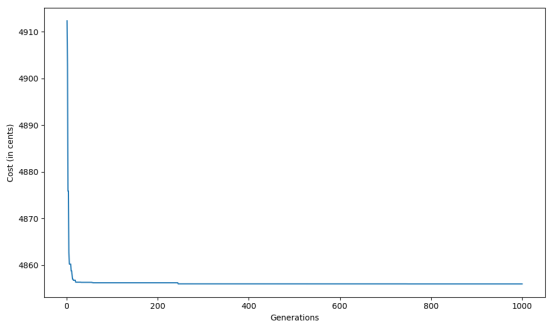

FIGURE 4. Average GA run.

the SoC of the ESS varies with hour interval i in a stochastic run of GA (with corresponding cost 4852.95 cents). It can be observed that the ESS is charged in the mornings (when the grid price is lower) and discharged in the later parts of the day (when the grid price is higher).

B. RESULTS USING PSO

For PSO, the recommended value for both the acceleration coefficients c_1 and c_2 is 2 [30], because on average it makes the weights for the social and cognition parts of Equation 1 equal to 1 (since r_1 and r_2 are real numbers in the range [0,1]). This ensures a balance between the local and global search for the optimal solution. We obtained the best results with $c_1 = c_2 = 2.025$. [49] showed that it's best to set initial particle velocities to zero or random values close to zero, so we set $v_{initial} = 0$. As for m_p, the population size, in the majority of applications, authors set m_p in between 20 and 50 particles [50]. However, we obtained the best results with

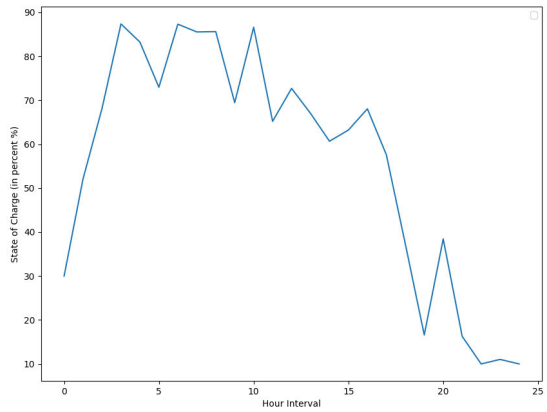

FIGURE 5. Hourly state of charge of battery using GA.

TABLE 5. Cost obtained using PSO with constant inertia factor.

Run	Cost (Cents)
1	4991.76
2	4854.47
3	4941.72
4	4926.05
5	4896.0
6	4873.11
7	4907.0
8	4965.78
9	4945.37
10	4929.03
Average	4923.03

a high population size of 1000 particles. We observed no additional benefits for $m_p > 1000$. As for the number of iterations, max_{iter}, we found no additional reduction in total cost for $max_{iter} > 1000$.

We initially set the inertia factor, w, at 0.9, which falls in the range recommended by [30] to find the global optimum. With these parameter values, the result was similar to that of GA, with the average cost over ten stochastic runs being 4923.03 cents (depicted in Table 5). This is an improvement of 10.02% over NPBA. Next, we used a linearly decreasing inertia factor, which was shown to generally perform better by [30]. In this scheme, the inertia factor, w, varies in each iteration according to Equation 3. This scheme worked best with $w_{min} = 0.5$ and $w_{max} = 1.4$. However, there was a negligible improvement over the constant inertia factor scheme, with the average cost obtained over 10 stochastic runs being 4880.82 cents (depicted in Table 6), an improvement of 10.79% over NPBA.

Then, we removed the momentum factor altogether, which was not found to be effective by the paper that first proposed PSO [12]. Surprisingly, however, when the inertia factor was made 0, the algorithm returned much better solutions overall, with the average cost obtained over 10 stochastic runs (depicted in Table 7) being 4688.68 cents, a 14.3% improvement over NPBA. This is over 3 percentage points better than GA and other PSO simulations with non-zero inertia factors. Moreover, it can be seen that without the momentum factor,

TABLE 6. Cost obtained using PSO with linearly decreasing inertia factor.

Run	Cost (Cents)
1	4938.74
2	4887.30
3	4928.73
4	4860.17
5	4885.27
6	4866.89
7	4868.59
8	4796.40
9	4930.74
10	4845.41
Average	4880.82

TABLE 7. Cost obtained using PSO without particle momentum.

Run	Cost (Cents)
1	4686.26
2	4686.26
3	4686.26
4	4709.02
5	4686.48
6	4686.40
7	4686.26
8	4687.32
9	4686.26
10	4686.26
Average	4688.68

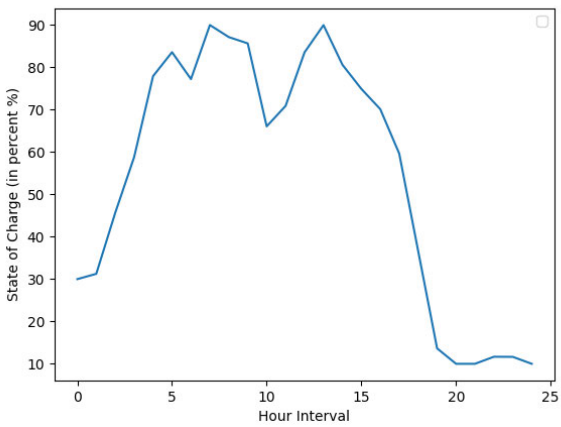

FIGURE 6. Hourly state of charge of battery using PSO.

PSO does not stagnate like GA and gradually moves closer to a better solution in each iteration, as depicted by Figure 7. Figure 6 depicts how the SoC of the ESS varies with hour interval i in a stochastic run of PSO (with corresponding cost 4686.33 cents).

The above results are for the 8-home microgrid. Next, we ran the algorithms on the 20-home and 40-home microgrids and found similar results, summarized in Table 8. Note that the last column Avg. Cost Reduction is an unweighted average of the percentage improvement over NPBA provided by the respective algorithm over all three data-sets.

It should be noted that the performance of our proposed algorithms is highly dependent on the parameters chosen. Parameter values that work well for our data-sets may or may not work well for other data-sets. Therefore, it's important to experiment with different parameter values and settle on what

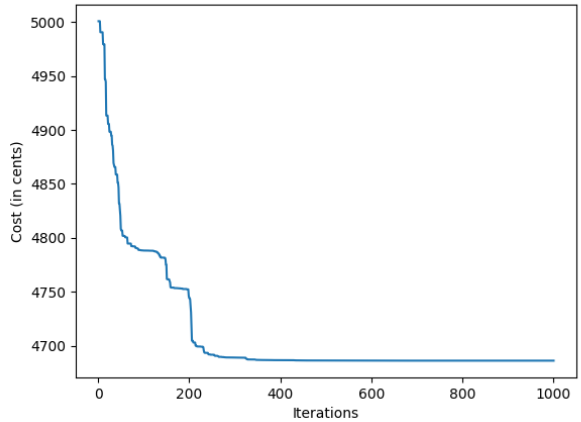

FIGURE 7. Average PSO run with no momentum.

TABLE 8. Comparison of costs.

Algorithm	Avg. Cost (8 Homes)	Avg. Cost (20 Homes)	Avg. Cost (40 Homes)	Avg. Cost Reduction Vs NPBA
GA	4865.27	12079.02	24302.36	11.31 %
PSO with constant w	4923.03	12269.95	24408.67	10.36%
PSO with linearly decreasing w	4880.82	12264.28	24540.84	10.47 %
PSO with $w = 0$	4688.68	11716.2	23446.65	14.31 %

works well for the data-set under consideration. Also, it's important to understand that meta-heuristics do not guarantee solution optimality. That said, they generally do return close-to-optimal solutions. It's up to the practitioner to tweak the parameters of the algorithm to obtain good solutions.

V. CONCLUSION

Taking into account the possibility of dynamic electricity pricing in the future, we set out to develop an algorithm that reduced the overall cost paid by consumers by scheduling the ESS in a microgrid optimally. We also considered constraints like the battery's maximum depth of charge and discharge to better emulate a real-world scenario. We first defined the problem statement mathematically. Then, we defined the search space and studied its structural properties. We proved that our search space has a magnification of at least $50 \times (B_c - B_d + 1)$, where B_c and B_d are the maximum depths of charge and discharge of the ESS respectively. Then, we described our algorithmic approach for this optimization problem by using Genetic Algorithm and Particle Swarm Optimization. We then applied the algorithms to forecast data obtained from [48]. We found that both GA and PSO fared well compared to the Net Power Based Algorithm. However, PSO with the momentum factor removed consistently provided the best cost reductions of over 14%. This reduction in cost over an extended span of time translates to a better incentive to shift to smart grids and incorporate renewable energy source infrastructure. We look forward to seeing the performance of other bio-inspired algorithms on the proposed search space.

REFERENCES

[1] M. I. Henderson, D. Novosel, and M. L. Crow, "Electric power grid modernization trends, challenges, and opportunities," in *Proc. IEEE Technol. Trend Paper*, 2017, pp. 1–17.

[2] X. Fang, S. Misra, G. Xue, and D. Yang, "Smart grid—The new and improved power grid: A survey," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 4, pp. 944–980, 4th Quart., 2012.

[3] H. Farhangi, "The path of the smart grid," *IEEE Power Energy Mag.*, vol. 8, no. 1, pp. 18–28, Jan. 2010.

[4] A. Banerji, D. Sen, A. K. Bera, D. Ray, D. Paul, A. Bhakat, and S. K. Biswas, "Microgrid: A review," in *Proc. IEEE Global Humanitarian Technol. Conf., South Asia Satell. (GHTC-SAS)*, Aug. 2013, pp. 27–35.

[5] D. T. Ton and M. A. Smith, "The U.S. department of Energy's microgrid initiative," *Electr. J.*, vol. 25, no. 8, pp. 84–94, Oct. 2012.

[6] C. Zhou, K. Qian, M. Allan, and W. Zhou, "Modeling of the cost of EV battery wear due to V2G application in power systems," *IEEE Trans. Energy Convers.*, vol. 26, no. 4, pp. 1041–1050, Dec. 2011.

[7] A. P. Sanghvi, "Flexible strategies for load/demand management using dynamic pricing," *IEEE Trans. Power Syst.*, vol. 4, no. 1, pp. 83–93, Feb. 1989.

[8] T. Logenthiran, D. Srinivasan, and E. Phyu, "Particle swarm optimization for demand side management in smart grid," in *Proc. IEEE Innov. Smart Grid Technol. Asia (ISGT ASIA)*, Nov. 2015, pp. 1–6.

[9] T. Logenthiran, D. Srinivasan, and T. Z. Shun, "Demand side management in smart grid using heuristic optimization," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1244–1252, Sep. 2012.

[10] D. Setlhaolo, X. Xia, and J. Zhang, "Optimal scheduling of household appliances for demand response," *Electr. Power Syst. Res.*, vol. 116, pp. 24–28, Nov. 2014.

[11] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. New York, NY, USA: Addison-Wesley, 1989.

[12] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE ICNN*, vol. 4, Nov./Dec. 1995, pp. 1942–1948.

[13] Y. Yang, J. Wu, Y. Chen, and C. Li, "A new strategy for short-term load forecasting," *Abstract Appl. Anal.*, vol. 2013, 2013, Art. no. 208964, doi: 10.1155/2013/208964.

[14] S. Skolthanarat, U. Lewlomphaisarl, and K. Tungpimolrut, "Short-term load forecasting algorithm and optimization in smart grid operations and planning," in *Proc. IEEE Conf. Technol. Sustainability (SusTech)*, Jul. 2014, pp. 165–171.

[15] S. Dutta, Y. Li, A. Venkataraman, L. M. Costa, T. Jiang, R. Plana, P. Tordjman, F. H. Choo, C. F. Foo, and H. B. Puttgen, "Load and renewable energy forecasting for a microgrid using persistence technique," *Energy Procedia*, vol. 143, pp. 617–622, Dec. 2017.

[16] J. Zheng, C. Xu, Z. Zhang, and X. Li, "Electric load forecasting in smart grids using Long-Short-Term-Memory based recurrent neural network," in *Proc. 51st Annu. Conf. Inf. Sci. Syst. (CISS)*, Mar. 2017, pp. 1–6.

[17] A. Ahmad, N. Javaid, A. Mateen, M. Awais, and Z. A. Khan, "Short-term load forecasting in smart grids: An intelligent modular approach," *Energies*, vol. 12, no. 1, p. 164, Jan. 2019.

[18] L. Youn and C. Sohyung, "Optimal operation of energy storage using linear programming technique," in *Proc. World Congr. Eng. Comput. Sci.*, in Lecture Notes in Computer Science, vol. 2178, Oct. 2009, pp. 480–485.

[19] D. K. Maly and K. S. Kwan, "Optimal battery energy storage system (BESS) charge scheduling with dynamic programming," *IEE Proc.-Sci., Meas. Technol.*, vol. 142, no. 6, pp. 453–458, Nov. 1995.

[20] R. Mallol-Poyato, S. Jiménez-Fernández, P. Díaz-Villar, and S. Salcedo-Sanz, "Joint optimization of a Microgrid's structure design and its operation using a two-steps evolutionary algorithm," *Energy*, vol. 94, pp. 775–785, Jan. 2016.

[21] C. D. Korkas, S. Baldi, I. Michailidis, and E. B. Kosmatopoulos, "Occupancy-based demand response and thermal comfort optimization in microgrids with renewable energy sources and energy storage," *Appl. Energy*, vol. 163, pp. 93–104, Feb. 2016.

[22] M. Marzband, M. Ghadimi, A. Sumper, and J. L. Domínguez-García, "Experimental validation of a real-time energy management system using multi-period gravitational search algorithm for microgrids in islanded mode," *Appl. Energy*, vol. 128, pp. 164–174, Sep. 2014.

[23] S. Esmaeili, A. Anvari-Moghaddam, and S. Jadid, "Optimal operational scheduling of reconfigurable multi-microgrids considering energy storage systems," *Energies*, vol. 12, pp. 1–21, May 2019.

[24] M. Marzband, A. Sumper, A. Ruiz-Álvarez, J. L. Domínguez-García, and B. Tomoiagă, "Experimental evaluation of a real time energy management system for stand-alone microgrids in day-ahead markets," *Appl. Energy*, vol. 106, pp. 365–376, Jun. 2013.

[25] C. D. Korkas, S. Baldi, and E. B. Kosmatopoulos, "Grid-connected microgrids: Demand management via distributed control and human-in-the-loop optimization," in *Advances in Renewable Energies and Power Technologies*, I. Yahyaoui, Ed. Amsterdam, The Netherlands: Elsevier, 2018, pp. 315–344.

[26] S. Esmaeili, A. Anvari-Moghaddam, and S. Jadid, "Optimal operation scheduling of a microgrid incorporating battery swapping stations," *IEEE Trans. Power Syst.*, vol. 34, no. 6, pp. 5063–5072, Nov. 2019.

[27] R. Sharifi, A. Anvari-Moghaddam, S. H. Fathi, and V. Vahidinasab, "A bi-level model for strategic bidding of a price-maker retailer with flexible demands in day-ahead electricity market," *Int. J. Electr. Power Energy Syst.*, vol. 121, Oct. 2020, Art. no. 106065.

[28] M. Vahedipour-Dahraie, H. Rashidizadeh-Kermani, A. Anvari-Moghaddam, and P. Siano, "Risk-averse probabilistic framework for scheduling of virtual power plants considering demand response and uncertainties," *Int. J. Electr. Power Energy Syst.*, vol. 121, Oct. 2020, Art. no. 106126.

[29] H. Lingaraj, "A study on genetic algorithm and its applications," *Int. J. Comput. Sci. Eng.*, vol. 4, pp. 139–143, Oct. 2016.

[30] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE Int. Conf. Evol. Comput. IEEE World Congr. Comput. Intell.*, May 1998, pp. 69–73.

[31] P. S. Oliveto and C. Witt, "Improved time complexity analysis of the simple genetic algorithm," *Theor. Comput. Sci.*, vol. 605, pp. 21–41, Nov. 2015.

[32] D. Antipov and B. Doerr, "Runtime analysis of a heavy-tailed $(1+(\lambda,\lambda))$ genetic algorithm on jump functions," in *Parallel Problem Solving From Nature—PPSN XVI* T. Bäck, M. Preuss, A. Deutz, H. Wang, C. Doerr, M. Emmerich, and H. Trautmann, Eds. Cham, Switzerland: Springer, 2020, pp. 545–559.

[33] B. Doerr, "An exponential lower bound for the runtime of the cGA on jump functions," 2019, *arXiv:1904.08415*. [Online]. Available: https://arxiv.org/abs/1904.08415

[34] D. Antipov, M. Buzdalov, and B. Doerr, "First steps towards a runtime analysis when starting with a good solution," in *Parallel Problem Solving from Nature—PPSN XVI* T. Bäck, M. Preuss, A. Deutz, H. Wang, C. Doerr, M. Emmerich, and H. Trautmann, Eds. Cham, Switzerland: Springer, 2020, pp. 560–573.

[35] D. Sudholt and C. Witt, "Runtime analysis of a binary particle swarm optimizer," *Theor. Comput. Sci.*, vol. 411, no. 21, pp. 2084–2100, May 2010.

[36] M. Mühlenthaler, A. Raß, M. Schmitt, and R. Wanka, "Exact Markov chain-based runtime analysis of a discrete particle swarm optimization algorithm on sorting and onemax," *CoRR*, vol. abs/1902.01810, pp. 1–44, Feb. 2019.

[37] Z. Hu, Y. Liu, Q. Su, and J. Huo, "A multi-objective genetic algorithm designed for energy saving of the elevator system with complete information," in *Proc. IEEE Int. Energy Conf.*, Dec. 2010, pp. 126–130.

[38] M. S. Ismail, M. Moghavvemi, and T. M. I. Mahlia, "Genetic algorithm based optimization on modeling and design of hybrid renewable energy systems," *Energy Convers. Manage.*, vol. 85, pp. 120–130, Sep. 2014.

[39] A. S. Pillai, K. Singh, V. Saravanan, A. Anpalagan, I. Woungang, and L. Barolli, "A genetic algorithm-based method for optimizing the energy consumption and performance of multiprocessor systems," *Soft Comput.*, vol. 22, no. 10, pp. 3271–3285, May 2018.

[40] O. E. Canyurt, H. K. Ozturk, A. Hepbasli, and Z. Utlu, "Genetic algorithm (GA) approaches for the transport energy demand estimation: Model development and application," *Energy Sour., A, Recovery, Utilization, Environ. Effects*, vol. 28, no. 15, pp. 1405–1413, Nov. 2006.

[41] Y.-Y. Hong and P.-S. Yo, "Novel genetic algorithm-based energy management in a factory power system considering uncertain photovoltaic energies," *Appl. Sci.*, vol. 7, no. 5, p. 438, Apr. 2017.

[42] M. Amer, A. Namaane, and N. K. M'Sirdi, "Optimization of hybrid renewable energy systems (HRES) using PSO for cost reduction," *Energy Procedia*, vol. 42, pp. 318–327, 2013.

[43] M. Sharafi and T. Y. ELMekkawy, "Multi-objective optimal design of hybrid renewable energy systems using PSO-simulation based approach," *Renew. Energy*, vol. 68, pp. 67–79, Aug. 2014.

[44] O. H. Mohammed, Y. Amirat, and M. Benbouzid, "Particle swarm optimization of a hybrid Wind/Tidal/PV/battery energy system. Application to a remote area in Bretagne, France," *Energy Procedia*, vol. 162, pp. 87–96, Apr. 2019.

[45] BS Power. (2018). *What is Microgrid*. [Online]. Available: https://www.blueskypower.com/wp-content/uploads/2018/01/What_is_Microgrid.png

[46] A. Sinclair, *Algorithms for Random Generation and Counting: A Markov Chain Approach*. Cambridge, MA, USA: Birkhauser Boston, 1993.

[47] K. B. Ajitha Shenoy, S. Biswas, and P. P. Kurur, "Efficacy of the metropolis algorithm for the minimum-weight codeword problem using codeword and generator search spaces," *IEEE Trans. Evol. Comput.*, vol. 24, no. 4, pp. 664–678, Aug. 2020.
[48] Z. Garroussi, "Data for demand-side management of multiple homes," Mendeley Data, V1, 2019, doi: 10.17632/hkyjg2spxf.1.
[49] A. Engelbrecht, "Particle swarm optimization: Velocity initialization," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2012, pp. 1–8.
[50] A. P. Piotrowski, J. J. Napiorkowski, and A. E. Piotrowska, "Population size in particle swarm optimization," *Swarm Evol. Comput.*, vol. 58, pp. 1–18, May 2020.

AJAY RAGHAVAN received the bachelor's degree in information technology from the Manipal Institute of Technology, in 2020. His research interests include algorithms and computer systems.

PAARTH MAAN received the B.Tech. degree in information technology with a minor in digital marketing from the Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, in 2020. He is inclined towards the development and application of various technical concepts. He has worked and held board positions with various tech and non-tech clubs. He was the Category Head during two college fests (TechTatva and Revels). He has been part of various sports team throughout his college years. He was the Head Boy and the Sports Captain during his senior years with the Sanskrit School, Pune. He is looking forward to gain experience and exposure in the information technology sector. His research interests include algorithms, computer systems, computer vision, evolutionary algorithms, and social network analysis.

AJITHA K. B. SHENOY received the M.Sc. degree in mathematics from Kannur University, in 1999, the M.Tech. degree in computer and information science from the Cochin University of Science and Technology, in 2003, and the Ph.D. degree in computer science from the Department of Computer Science and Engineering, IIT Kanpur, in 2015. He is currently working as an Associate Professor with the Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal. His research interests include algorithm, computational complexity theory, randomized local search heuristics, evolutionary algorithms, computer vision, and data compression.