Federated Swarm Optimization for Privacy-Preserving Parameter Tuning in Networked Learning Systems

Usman Javed[1]

[1] Mehran University of Engineering and Technology, Department of Telecommunication Engineering, Jamshoro–Hyderabad Road, Jamshoro, Sindh, Pakistan

ABSTRACT

Networked learning systems increasingly operate over distributed data sources that cannot be centralized due to regulatory, organizational, or technical constraints. At the same time, the performance of such systems is sensitive to parameter choices at both the model and system levels, including learning rates, regularization strengths, aggregation weights, and communication schedules. Conventional parameter tuning procedures that rely on centralized access to validation data or repeated global retraining are often incompatible with privacy and communication constraints. Swarm-based metaheuristics provide a flexible mechanism for exploring high-dimensional parameter spaces, yet standard designs assume globally visible objective evaluations. This paper examines a federated swarm optimization strategy for parameter tuning in networked learning systems where data remain local and only carefully controlled summaries are exchanged. The approach conceptualizes each client as an autonomous optimizer that evaluates candidate parameter configurations on local tasks while contributing to a collective search process mediated by a coordinating server. Privacy is addressed by combining aggregation mechanisms with controlled perturbations of shared statistics, while communication costs are managed by sparse and event-driven exchanges of swarm information. The study presents a formal problem formulation, an algorithmic design integrating federated interaction patterns with swarm dynamics, and an analysis of privacy and convergence properties under simplified assumptions on local loss functions and network connectivity. Numerical considerations relevant to practical deployment, such as dimensionality, parameter constraints, and sensitivity to hyperparameters of the swarm itself, are discussed as part of a broader examination of the applicability of federated swarm optimization to privacy-aware networked learning.

KEYWORDS

1. Introduction

Distributed and networked learning systems have become a common architectural choice in environments where data are generated at the edge of the network, for example on embedded devices, mobile phones, or institutional data silos [1]. In such settings, learning algorithms must operate under constrained communication, heterogeneous computation capabilities, and restrictive privacy or governance rules. A recurring challenge in these systems is the tuning of model and algorithm parameters that strongly influence predictive accuracy, robustness, and resource usage [2]. While centralized machine learning typically relies on cross-validation and grid or random search over a parameter space, these techniques are difficult to deploy when validation data cannot be pooled and when repeated retraining or evaluation incurs substantial communication and computation overhead.

This situation has motivated interest in optimization and learning paradigms that can coordinate computations across multiple clients without exposing raw local data. Federated learning is an example of such a paradigm, where clients compute local model updates that are aggregated by a central server [3]. In many practical implementations, however, the param-

Article info:
Usman Javed. *Federated Swarm Optimization for Privacy-Preserving Parameter Tuning in Networked Learning Systems*. Nextqueries. Publishing

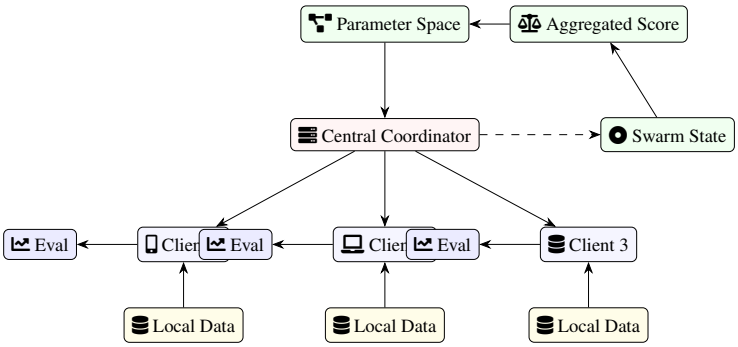

Figure 1 Federated swarm optimization architecture with distributed clients hosting local data, a central coordinator maintaining global swarm state, and a shared parameter space where aggregated scores inform the evolution of candidate configurations.

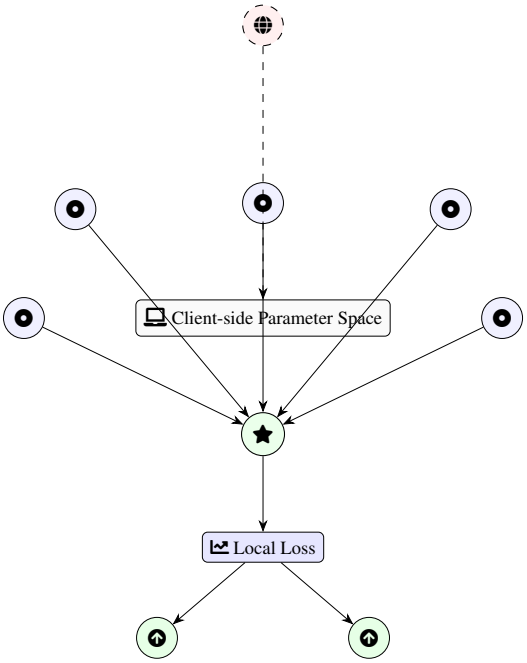

Figure 2 Local view of swarm particles at a single client, where candidate parameter positions evolve toward the local best configuration while being weakly influenced by a global reference shared across the network.

eters of the underlying learning algorithm, including learning rates, momentum coefficients, and regularization terms, are still chosen by heuristic procedures or transferred from centralized prototypes. When data distributions across clients differ, and when privacy constraints preclude rich monitoring of validation performance, the choice of these parameters becomes nontrivial and can lead to suboptimal performance or unstable training dynamics.

Swarm-based optimization methods, such as variants of particle swarm optimization, have been used in centralized settings for continuous parameter tuning in high-dimensional spaces [4]. These methods are attractive in part due to their derivative-free nature and their ability to explore multimodal landscapes. They rely on a population of candidate solutions that move through the parameter space according to rules that combine local exploration with attraction toward historically successful configurations [5]. Classical formulations assume that each candidate solution can be evaluated on a globally defined objective, or at least that information about the objective values can be shared without restriction. In a networked learning environment with local validation data, this assumption no longer holds.

Federated swarm optimization adapts swarm-based search procedures to a context in which each client maintains and evaluates candidate parameter configurations using only its local data [6]. A coordinating server aggregates selected information about candidate performance across the network, forming a global or partially global notion of quality that can influence the motion of the swarm. Because raw data are not transmitted, the aggregated signals become the primary mechanism through which the distributed system aligns its search for suitable parameter values. This setup introduces new trade-offs between exploration and exploitation, between privacy and informativeness of shared signals, and between communication cost and search efficiency [7].

Privacy-preserving parameter tuning in this context requires more than simply withholding raw data. Evaluations of pa-

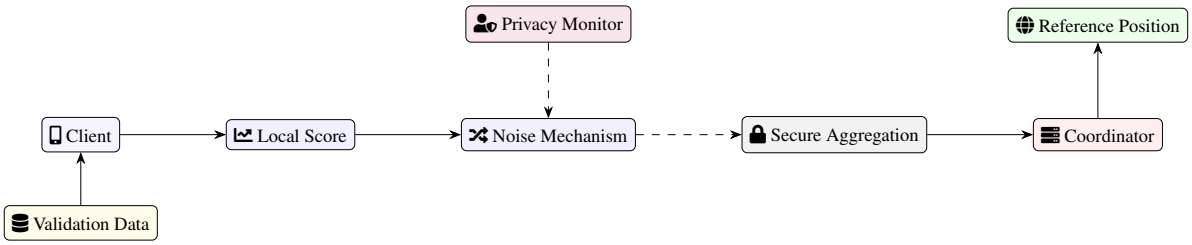

Figure 3 Privacy-preserving reporting pipeline in which a client converts local validation data into a score, perturbs the score through a randomized mechanism, transmits the result through secure aggregation, and contributes to the global reference parameter maintained at the coordinator.

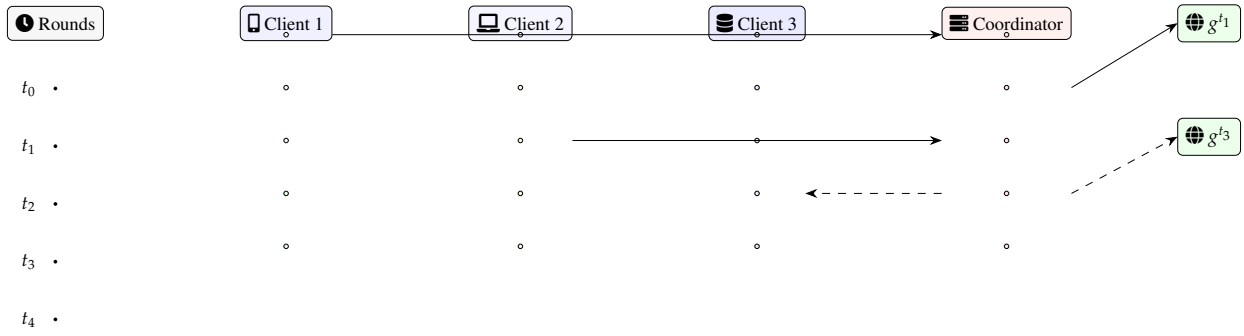

Figure 4 Round-based communication schedule illustrating client activity over time, selective uploads from clients to the coordinator, and intermittent updates of the global reference parameter shared across the network.

rameter configurations on local validation sets may themselves leak information if reported without protection, especially when combined across multiple rounds or with side information about client distributions [8]. To mitigate such risks, aggregation mechanisms can be combined with noise injection strategies that provide formal privacy guarantees while maintaining usable optimization signals. The calibration of these mechanisms interacts with the design of the swarm dynamics, since the effective signal-to-noise ratio influences convergence behavior and the stability of the population of candidate solutions.

This paper develops a formal framework for federated swarm optimization in networked learning systems with privacy requirements [9]. The framework models clients as agents connected through a coordinating entity, with each agent holding local data and participating in a collective optimization process over a shared parameter space. The swarm, distributed across clients, evolves according to update rules that rely on locally computed loss values and globally aggregated quality indicators. Privacy is modeled through constraints on the information that may be revealed about local loss functions, represented as restrictions on the form and precision of the communicated quantities [10]. Within this setting, the paper describes an algorithmic template, examines structural properties of the resulting dynamics, and discusses implications for communication cost, privacy loss, and practical deployment in heterogeneous networks.

2. Networked Learning and Parameter Tuning Model

The starting point for the analysis is a networked learning system composed of a finite set of clients indexed by a set K, with $|K| = m$ [11]. Each client k holds a local dataset that is not shared with any other entity, including a coordinating server. The learning task is modeled through a parameter vector that resides in a Euclidean space of dimension d. Denote this parameter vector by w, where $w \in \mathbb{R}^d$. The parameter may represent model weights, algorithmic hyperparameters, or a mixture of both, depending on the specific application [12].

For each client k, a local loss function is defined as a mapping from the parameter space to the real numbers. This mapping is denoted by F_k, where

$$F_k(w) = \mathbb{E}_{z \sim D_k}\big[\ell(w,z)\big].$$

Here D_k denotes the local data distribution at client k, and ℓ is a per-sample loss function. The expectation is taken with respect to the randomness in the local data [13]. In practice, the loss is approximated by empirical averages over finite datasets, but for modeling purposes it is convenient to express it in expectation form.

The global performance of a parameter vector across the network is modeled as an aggregation of the local loss functions. A simple choice, which is adopted here for analytical clarity, is a weighted average [14]. Let α_k be nonnegative weights that sum to one. The global loss is then

$$F(w) = \sum_{k \in K} \alpha_k F_k(w).$$

The parameter tuning task is to identify values of w that minimize this global loss [15]. In cases where local distributions differ substantially, the global loss captures a compromise between clients, moderated by the weights α_k.

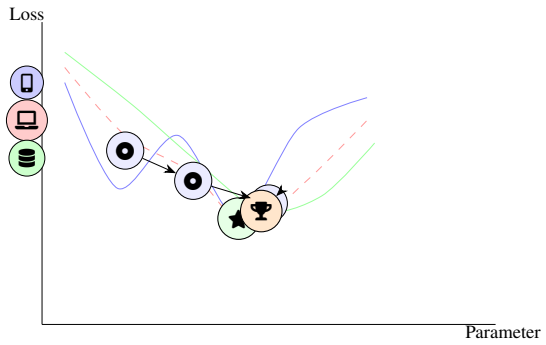

Figure 5 One-dimensional illustration of a shared parameter axis with client-specific loss surfaces, showing swarm particles migrating toward locally and globally favorable regions of the landscape as the swarm integrates information from heterogeneous clients.

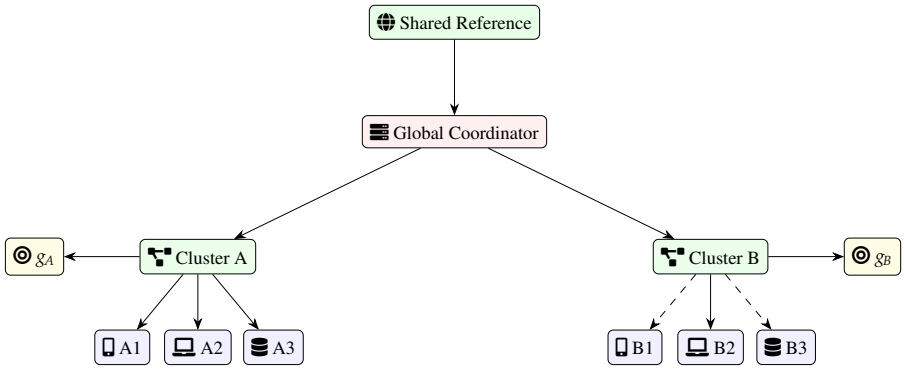

Figure 6 Hierarchical deployment of federated swarm optimization where clients are grouped into clusters with local aggregators that maintain cluster-specific reference parameters, while a global coordinator reconciles cluster summaries into a shared reference.

The learning algorithm that uses the tuned parameters is treated as a black box that, given a parameter vector w, produces a model or a sequence of updates that can be evaluated on local validation data. In many applications, the performance measure used for tuning is derived from the same loss ℓ as used during training, but evaluated on a separate validation set. The distinction between training and validation is not critical for the modeling here; what matters is that each client can compute a scalar performance metric $G_k(w)$ that reflects the quality of w from the perspective of client k. For simplicity, one may set $G_k = F_k$, although in practice more complex metrics can be used.

In a classical centralized search procedure over w, a central controller would propose parameter vectors, obtain evaluations of G_k from all clients or from a pooled dataset, and use these evaluations to guide the search. In a privacy-constrained networked setting, direct reporting of $G_k(w)$ to a central server may be restricted or must be perturbed to satisfy privacy policies. The optimization process must therefore rely on limited, noisy, or aggregated information about the local performance of candidate parameters [16].

Privacy requirements can be formalized in different ways. A common formulation is based on the notion of indistinguishability between neighboring datasets. For each client k, consider local datasets that differ by at most one sample [17]. A mechanism that maps these datasets to shared statistics is said to satisfy a privacy constraint if its output distributions on neighboring datasets are close in a specified sense. In the context of federated swarm optimization, the reported statistics include evaluations of candidate parameters and possibly other summary quantities derived from local computations [18]. To protect privacy, noise may be added to these quantities before they are transmitted, and aggregation mechanisms can be structured so that individual contributions cannot be isolated by an adversary that observes network traffic or server states.

Another dimension of the model concerns the network topology and communication pattern. While a star topology with a central server that communicates with all clients is common in federated learning, other structures are possible, including peer-to-peer connectivity or hierarchical clusters [19]. For the present discussion, it is assumed that a central coordinating entity exists and that each client can exchange messages with this entity, although the frequency and content of these messages are constrained. The communication pattern unfolds in rounds, indexed by a nonnegative integer t, during which clients may send summaries of local evaluations and receive updated information about the swarm state.

It is convenient to introduce notation for the state of the swarm that explores the parameter space [20]. Let P denote the total number of particles, or candidate parameter configurations, in the swarm. These particles are distributed across the clients [21]. For client k, let P_k be the set of indices of particles managed by that client. The cardinalities $|P_k|$ may vary across clients, and their sum equals P. Each particle i is associated

with a parameter vector $x_i \in \mathbb{R}^d$ and an auxiliary state such as a velocity vector $v_i \in \mathbb{R}^d$. The mapping between particle indices and client identifiers is fixed throughout the optimization process or may evolve over time if particles migrate between clients.

The parameter tuning problem is then reinterpreted as guiding the swarm so that, over successive rounds, many particles concentrate near regions of the parameter space in which the global loss $F(w)$ is low. Because $F(w)$ is not directly observable at any single site, the swarm dynamics must be based on partially observed or aggregated information obtained from clients [22]. This setup differentiates the federated swarm scenario from centralized swarm optimization and motivates the algorithmic modifications described in subsequent sections.

3. Federated Swarm Optimization Algorithm

The federated swarm optimization algorithm considered in this paper adapts classical swarm dynamics to a setting with distributed data and constrained information exchange. The core idea is that each client maintains a subset of the swarm and evolves the associated particles based on local evaluations of their performance, while periodically communicating selected summaries to a central coordinator [23]. The coordinator aggregates this information to construct a notion of global or shared best configurations, which are then transmitted back to clients to influence further evolution of the swarm.

Each particle i is characterized at round t by its position x_i^t in the parameter space and by a velocity v_i^t. A standard form of update in swarm optimization can be expressed as

$$v_i^{t+1} = \omega^t v_i^t + c_1^t r_1^t (p_i^t - x_i^t) + c_2^t r_2^t (g_i^t - x_i^t),$$

followed by [24]

$$x_i^{t+1} = x_i^t + v_i^{t+1}.$$

Here p_i^t denotes the best position encountered so far by particle i according to a local performance measure, and g_i^t denotes a reference position that captures information about the performance of other particles. The coefficients ω^t, c_1^t, and c_2^t are scalar parameters that may vary with t, and r_1^t, r_2^t are random variables, often sampled independently from a uniform distribution on an interval. These equations are compact and remain within a limited width by design.

In the federated setting, the interpretation of p_i^t and g_i^t must be adapted. For each client k, the local best position for particle $i \in P_k$ is defined using evaluations of a performance measure G_k. At round t, client k evaluates the current position x_i^t on its validation data and obtains a scalar value $G_k(x_i^t)$. The local best position p_i^t is updated whenever the new evaluation is better than the value recorded at the previous local best. Because the evaluation uses only local data, the definition of p_i^t depends on the specific client managing particle i.

The role of g_i^t in the update equations is to introduce information about promising regions discovered by other particles, potentially at other clients. The simplest version of federated swarm optimization uses a single global reference position g^t that is shared across all particles. This global reference is derived from aggregated information collected at the server [25]. At the end of each round, clients transmit summaries of their local particle performance to the server. For example, for each client k, a candidate position q_k^t and its associated performance score can be selected among the particles in P_k. The server then identifies a position g^t whose associated aggregated score is better than or comparable to others and broadcasts this position to all clients. In this case, one sets $g_i^t = g^t$ in the velocity update for all particles.

To reduce communication, clients may not transmit information about all particles in each round. Instead, an event-driven strategy can be adopted in which a client transmits an update only when the performance of some local particle improves by more than a threshold [26]. Another possibility is to subsample a subset of particles or to quantize the positions before transmission, reducing the amount of information sent. These design choices introduce approximations into the global view of the swarm held by the server, but they can substantially reduce communication costs.

The federated algorithm proceeds in rounds [27]. At the beginning of round t, each client possesses the positions and velocities of its particles and the current reference position g^t received from the server. The client evaluates the performance of each local particle position using its local data, updates the local best positions p_i^t, and then computes new velocities and positions using the update equations. After updating, the client may choose one or more candidate positions to report to the server, along with associated performance metrics that are possibly perturbed to satisfy privacy constraints. The server aggregates these reports to update the global reference position g^{t+1}. This cycle repeats until a termination condition is reached, such as a maximum number of rounds or stabilization of the swarm [28].

Parameter constraints can be incorporated into the swarm dynamics by projecting updated positions onto a feasible set. Suppose the parameter space of interest is a closed and convex subset W of $\mathbb{R}^d$. The projection operator onto W is denoted by Π_W. After computing x_i^{t+1}, the client replaces it by

$$x_i^{t+1} = \Pi_W(x_i^t + v_i^{t+1}).$$

This projection can enforce bounds on individual coordinates, linear constraints, or more complex structural restrictions, provided they can be expressed as a convex set with an efficiently computable projection.

The algorithmic description above leaves several degrees of freedom open, including the choice of coefficients ω^t, c_1^t, c_2^t, the distribution of random terms, the criteria for selecting candidate positions to send to the server, and the rules by which the server aggregates reported positions into a reference g^t. These degrees of freedom can be used to adapt the algorithm to specific applications, to account for heterogeneous client capacities, or to manage privacy and communication budgets [29]. The next section extends the algorithmic template by incorporating explicit privacy-preserving mechanisms into the communication protocol.

s

Parameter	Symbol	Typical Range	Role
Inertia coefficient	ω^t	$[0.3, 0.9]$	Controls momentum of particle motion
Cognitive weight	c_1^t	$[0.5, 2.5]$	Attraction to particle-local best
Social weight	c_2^t	$[0.5, 2.5]$	Attraction to global reference
Swarm size	P	$[16, 256]$	Number of candidate configurations
Dimension	d	$[16, 10^3]$	Length of parameter vector w
Max rounds	T	$[50, 500]$	Upper bound on optimization iterations

Table 1 Core swarm hyperparameters used in federated parameter tuning together with representative ranges and qualitative roles in the dynamics.

Quantity	Client-side Cost	Server-side Cost	Scaling Factor
Local evaluation $G_k(x)$	$\mathcal{O}(n_k d)$	none	grows with data size
Velocity update v_i^t	$\mathcal{O}(d)$	none	per particle
Position update x_i^t	$\mathcal{O}(d)$	none	per particle
Projection $\Pi_W(x)$	$\mathcal{O}(d)$	none	depends on constraint set
Aggregation of scores	none	$\mathcal{O}(m)$	linear in clients
Reference update g^t	none	$\mathcal{O}(Pd)$	depends on selection rule

Table 2 Computational complexity of principal operations in federated swarm optimization, separated by client-side and server-side contributions.

4. Privacy-Preserving Mechanisms

In federated swarm optimization, privacy concerns arise from the transmission of information about local evaluations of parameter configurations. Even if raw data are not shared, repeated reporting of precise performance metrics can enable reconstruction or inference attacks, especially when an adversary has auxiliary knowledge about client distributions or has access to the sequence of messages over many rounds [30]. To reduce such risks, privacy-preserving mechanisms can be integrated into the communication process between clients and the server.

A common formal framework for studying privacy is based on randomization of the reported outputs [31]. For each client k, consider a reporting mechanism that takes as input the local dataset and a set of candidate positions and produces as output one or more messages to be sent to the server. The mechanism is randomized in the sense that, for a fixed input, it may produce different outputs on different executions. The randomness is used to limit the extent to which the output reveals information about individual data points [32]. In the present setting, the inputs to the mechanism include loss values $G_k(x)$ computed on the local data for various positions x. These loss values, or derived scores, are then perturbed before transmission.

A simple mechanism for perturbation is additive noise. Suppose client k wishes to report a scalar score s_k^t associated with a candidate position $\hat{x}_k^t$ at round t. The client constructs a noisy score [33]

$$\tilde{s}_k^t = s_k^t + \xi_k^t,$$

where ξ_k^t is a random variable drawn from a distribution that is independent of the data, such as a Gaussian or Laplace distribution with mean zero. The variance or scale of ξ_k^t is chosen based on a sensitivity bound that quantifies how much the score s_k^t could change when a single data point in the local dataset is modified. Under appropriate calibrations, the resulting mechanism can satisfy a formal privacy guarantee for each client across multiple rounds.

The presence of noise in the reported scores affects the server's ability to identify high-quality candidate positions. When the server receives reports $\tilde{s}_k^t$ from multiple clients, it must form an estimate of the relative quality of the associated positions $\hat{x}_k^t$. A simple strategy is to treat the noisy scores as unbiased estimates and select the position with the smallest $\tilde{s}_k^t$ as the new reference g^{t+1}. Alternatively, the server can maintain a distribution over candidate positions and update it in a probabilistic manner based on the noisy scores, thereby smoothing the impact of any single noisy report [34].

To further reduce the risk of linking reported scores to individual clients, secure aggregation mechanisms can be employed. In such mechanisms, each client masks its report using random values shared with other clients in such a way that the server can recover only an aggregate, such as a sum or average of the scores, but cannot see individual contributions [35]. For example, clients can collaborate to generate pairwise random masks that cancel out when summed across clients. If the server receives only aggregated noisy scores

$$\bar{s}^t = \sum_{k \in K} \tilde{s}_k^t,$$

then individual client scores remain hidden, although this form of aggregation is more naturally suited for averaging than for identification of a single best candidate [36]. To adapt aggregation to the swarm setting, one can aggregate statistics that reflect distributions over candidate positions instead of individual scores.

An alternative is to let each client maintain a local ranking of its particles and to report only coarse information, such as the index of a locally best particle or a binary indicator of improvement. These reports can be encoded in a way that allows the server to infer global trends without observing precise loss values [37]. For instance, clients may use randomized response mechanisms to decide whether or not to signal that a candidate has improved beyond a local threshold, adding uncertainty to any inference about true performance while still providing useful hints for guiding the swarm.

The design of privacy-preserving mechanisms must be coordinated with the desired privacy budget across rounds [38]. If

Mechanism	Noise Distribution	Typical Scale	Comment
Additive Gaussian	$\mathcal{N}(0,\sigma^2)$	$\sigma \in [0.01, 0.5]$	Smooth perturbations of scores
Additive Laplace	$\mathrm{Lap}(0,b)$	$b \in [0.01, 0.3]$	Suitable for ℓ_1 sensitivity bounds
Randomized response	two-point	flip prob. $\in [5\%, 30\%]$	Coarse binary signaling
Score quantization	discrete grid	step $\in [10^{-3}, 10^{-1}]$	Reduces precision before noise
Secure aggregation	mask sum	implicit	Hides individual contributions

Table 3 Privacy-preserving reporting mechanisms used at clients before transmission of local statistics to the coordinator.

Strategy	Upload Pattern	Downlink Pattern	Bandwidth Use
Full synchronous	every round	every round	highest
Event driven	on improvement	periodic	reduced, adaptive
Subsampled particles	fixed subset	periodic	proportional to subset size
Clustered reporting	via aggregators	periodic	amortized over clusters
Sparse broadcast	compressed g^t	rare	heavily reduced downlink

Table 4 Representative communication strategies for exchanging swarm-related information between clients and coordinator under bandwidth constraints.

each round consumes a portion of a global privacy budget for each client, then the depth of the optimization process and the frequency of reporting must be balanced against the level of privacy protection. This introduces a trade-off: more frequent and precise reports can improve the efficiency of the optimization but may increase the cumulative privacy loss, whereas infrequent and heavily perturbed reports protect privacy at the cost of slower convergence or higher variance in the optimization trajectory.

From an analytical perspective, the effect of noise and limited precision on the swarm dynamics can be captured by modeling the reference position g^t as a random variable whose distribution depends on the true local performance of particles and on the noise mechanisms. For a given particle i, the update of its velocity at round t depends on g_i^t, which may be set equal to g^t or to a client-specific reference. If the noise variance is large relative to the variation in true scores, the direction and magnitude of the attraction toward the reference may become erratic, potentially reducing the algorithm's ability to exploit true improvements in performance [39]. Conversely, moderate noise can preserve the global direction of improvement while hiding individual contributions.

In practical deployments, privacy-preserving mechanisms can be adjusted dynamically based on monitoring of optimization progress and privacy consumption. For example, larger noise levels can be used in early rounds when the swarm is still exploring coarse regions of the parameter space, while smaller noise levels can be reserved for later rounds when fine-grained adjustments are necessary [40]. Similarly, the number of particles or the frequency of reporting can be reduced as the search narrows to a region of interest, conserving privacy budget and communication resources while focusing computational effort on refinement.

5. Convergence and Complexity Analysis

The convergence behavior of federated swarm optimization differs from classical centralized swarm optimization because of the additional sources of uncertainty and delay introduced by distributed evaluations, noisy reporting, and constrained communication [41]. A complete theoretical analysis under realistic conditions is challenging, especially when local loss functions are nonconvex and when client participation is intermittent. Nevertheless, useful insights can be obtained by examining simplified models that capture key aspects of the dynamics.

Consider a scenario in which all clients participate in every round, and where local loss functions are smooth and share a common minimizer $w^\star$ in the parameter space. In such a case, the global loss function admits the same minimizer, and one can analyze the behavior of the swarm in a neighborhood of $w^\star$. For each particle i, define the error vector [42]

$$e_i^t = x_i^t - w^\star.$$

Under certain parameter choices for the swarm dynamics, including bounds on the inertia coefficient and attraction coefficients, it is possible to show that the expected norm of e_i^t decreases over time, provided the noise in the reference position is not too large. This type of analysis builds on linearization of the update equations around the equilibrium and on bounding the spectral radius of an associated transition matrix.

To illustrate the structure, suppose the reference position g^t is equal to $w^\star$ for all t, and that noise in the updates arises only from the random coefficients r_1^t and r_2^t. The velocity update can then be written in terms of the error as

$$v_i^{t+1} = \omega^t v_i^t - c_1^t r_1^t e_i^t - c_2^t r_2^t e_i^t.$$

Combining this with the position update yields a linear stochastic system in the variables e_i^t and v_i^t. Conditions for mean-square stability of such systems translate into bounds on the coefficients and their expectations [43]. When the reference is not exactly at $w^\star$ but fluctuates around it due to noise in the reporting and aggregation process, additional terms appear that act as persistent disturbances in the error dynamics, leading to convergence to a neighborhood of $w^\star$ rather than to the exact minimizer.

The impact of communication constraints can be modeled by introducing delays and partial updates in the reference position. For example, if the server updates g^t only every L rounds, then for intermediate rounds clients use a stale reference. Analysis of the resulting dynamics can be based on viewing the swarm as operating under piecewise constant references, with occasional adjustments [44]. Stability and convergence rates then depend

Client Type	Data Size n_k	Loss Curvature	Participation Pattern
Edge sensor	small	low	intermittent
Mobile device	medium	moderate	bursty
Institutional node	large	high	regular
Archive server	very large	mixed	scheduled
Simulated client	configurable	configurable	synthetic schedule

Table 5 Categories of clients considered in networked learning scenarios and qualitative characteristics that influence federated swarm dynamics.

Regime	Coefficients	Swarm Behavior	Typical Use
Exploration dominated	ω high, c_1, c_2 small	broad search	early rounds
Local refinement	ω moderate, c_1 moderate	focus around local best	mid rounds
Consensus seeking	ω small, c_2 large	strong pull to g^t	late rounds
Noisy aggregation	c_2 moderate, noisy g^t	stochastic drift	high privacy
Decoupled clients	$c_2 \approx 0$	independent search	baseline comparison

Table 6 Qualitative convergence regimes induced by different choices of swarm coefficients and levels of aggregation noise.

on the frequency of updates and on how rapidly the swarm reacts to changes in the reference. Broadly, less frequent updates reduce communication but may slow convergence and can lead to oscillatory behavior if the swarm overreacts to outdated references.

Complexity analysis focuses on the computational and communication costs associated with each round and with the overall optimization process [45]. At each client k, the dominant computational cost arises from evaluating the loss function $G_k(x)$ for each local particle. If client k manages $|P_k|$ particles and uses a validation set of size n_k, then the cost per round of loss evaluation scales with $|P_k|n_k$, modulo the cost of a single evaluation of ℓ. Additional computational cost is due to the velocity and position updates, which scale with $|P_k|d$, and to any projection operations or local ranking computations.

Communication complexity is more sensitive to design choices. If each client were to send the position and precise score of each local particle to the server in every round, the communication load would be linear in Pd per round in each direction, which can be prohibitive for high-dimensional parameter spaces or large swarms. The federated algorithm therefore relies on sparse and possibly quantized communication, with clients sending only selected positions and coarse or noisy scores [46]. If each client sends a fixed number of candidate positions per round, say one or a small constant number, the communication load per round becomes independent of the swarm size and depends only on the number of clients and the dimensionality d. Quantization can further reduce the number of transmitted bits per coordinate.

From a complexity standpoint, one can also consider the number of rounds required to reach a parameter configuration whose global loss is within a target tolerance of the infimum [47]. This number depends on the condition of the loss landscape, the swarm parameters, the degree of noise, and the communication pattern. While precise bounds are difficult to derive, qualitative trends are as expected: larger swarms and more frequent communication tend to improve exploration and convergence at the cost of higher computation and communication, while stronger privacy protections that introduce more noise tend to slow convergence and may limit the attainable accuracy [48].

An additional aspect of the analysis concerns robustness to client heterogeneity. In many networked learning systems, clients have different data distributions, computational capabilities, and availability patterns. Heterogeneity in data distributions can be modeled by allowing the local loss functions F_k to have different minimizers. In such cases, a single global parameter vector may not be optimal for all clients, and the swarm may converge to a compromise solution that balances performance across the network [49]. The behavior of the federated swarm under such heterogeneity can be studied by examining how local evaluations influence the aggregated reference and how the swarm distributes itself in the parameter space.

Robustness to participation heterogeneity can be modeled by allowing clients to drop out or join at different rounds. When participation is intermittent, some particles may remain inactive for several rounds, while others continue to evolve [50]. The server must adapt its aggregation process to account for varying participation, possibly by weighting reports based on recency or by maintaining separate reference positions for groups of clients. These mechanisms further complicate the analysis but reflect realistic operating conditions [51].

6. Experimental Design and Discussion

Although the preceding sections focus on modeling and analysis, it is useful to outline how federated swarm optimization can be evaluated empirically in networked learning scenarios. Such evaluation typically involves constructing synthetic or semi-synthetic environments in which multiple clients hold local datasets, implementing the federated swarm algorithm with privacy-preserving mechanisms, and comparing its parameter tuning performance to that of baseline methods under various constraints.

A basic experimental setting can be constructed as follows [52]. The parameter vector w represents the weights of a linear model or a low-depth neural network used for prediction tasks on data distributed across clients. Local datasets can be derived from a common source by partitioning examples according to features or labels, thereby inducing heterogeneity in local distributions. For each client, a portion of its data is reserved for training the model under candidate parameters, and a separate portion is reserved for evaluating the performance metric G_k used in the swarm optimization. The federated swarm algo-

Objective	Notation	Example Constraint	Evaluation Site
Predictive loss	$F(w)$	$F(w) \leq \tau$	aggregated over clients
Communication cost	$C(w)$	$C(w) \leq C_{\max}$	coordinator statistics
Energy budget	$E_k(w)$	$E_k(w) \leq E_k^{\max}$	per client
Fairness deviation	$\Delta(w)$	$\Delta(w) \leq \gamma$	across-client disparity
Model size	$S(w)$	$S(w) \leq S_{\max}$	global parameter representation

Table 7 Multiobjective view on federated parameter tuning, listing representative quantities and example constraints considered during optimization.

Configuration	Swarm Size P	Rounds T	Privacy Noise Level
Lightweight	32	80	low
Balanced	64	150	medium
Communication aware	48	200	medium high
Privacy focused	80	250	high
High accuracy	128	300	low medium

Table 8 Example experimental configurations illustrating trade-offs between swarm size, optimization depth, and privacy noise in federated parameter tuning.

rithm is then applied to tune parameters such as regularization strengths, learning rates, or architectural coefficients, while respecting a budget on the number of communication rounds and on privacy loss [53].

To simulate privacy-preserving reporting, additive noise is applied to local scores or to aggregated statistics at the client side before transmission. The noise variance is chosen according to a target privacy level [54]. The resulting noisy scores are aggregated at the server to update the reference position g^t. Baseline methods for comparison can include centralized swarm optimization using pooled validation data, if such pooling is allowed in a simulated environment, and local tuning methods where each client tunes its parameters independently without coordination. The performance of the federated swarm can then be measured in terms of the global loss $F(w)$ achieved by the tuned parameters, communication cost, and privacy consumption.

Interpreting experimental results requires attention to how heterogeneity and privacy mechanisms interact with the swarm dynamics [55]. For example, when local data distributions differ significantly, a parameter configuration that performs well on one client may not perform well on another. The federated swarm aggregates these local perspectives through the reference position, which tends to favor configurations that balance performance across clients. Experiments can explore how different weighting schemes for clients, reflected in the choice of α_k, influence the final tuned parameters. In some applications, it may be desirable to give higher weight to clients with larger datasets or to those representing critical use cases [56].

Another dimension of experimental investigation concerns the choice of swarm hyperparameters, including the number of particles, the coefficients ω^t, c_1^t, c_2^t, and the strategies for updating them over time. These hyperparameters influence the trade-off between exploration and exploitation. Experiments can examine how different choices affect convergence speed, stability, and sensitivity to noise. For instance, larger inertia coefficients can encourage broader exploration early in the optimization but may lead to overshooting or oscillations unless reduced over time [57]. Similarly, strong attraction toward the reference position can accelerate convergence when the reference is reliable but can amplify the effect of noise when privacy-preserving perturbations are large.

Communication strategies are also amenable to empirical study [58]. Event-driven reporting, in which clients transmit updates only when local improvements exceed a threshold, can substantially reduce communication in environments where progress is incremental. Experiments can investigate how to set such thresholds as functions of local variance in loss evaluations or as adaptive quantities that change over the course of training. Quantization of transmitted positions and scores introduces additional approximation errors, but may be necessary to meet bandwidth constraints [59]. Measuring the impact of different quantization levels on the quality of tuned parameters helps inform trade-offs between precision and communication cost.

From a privacy perspective, empirical evaluation can track how noise in reported statistics impacts both optimization performance and privacy guarantees. While formal privacy analysis provides theoretical bounds, empirical observations can reveal how much noise can be tolerated in practice before optimization performance degrades [60]. Experiments can vary noise levels while monitoring convergence behavior, final loss values, and robustness to initialization. In addition, different privacy mechanisms, such as additive noise, randomized response, or secure aggregation of coarse indicators, can be compared in terms of the balance they offer between privacy protection and optimization efficacy [61].

Finally, the interpretability and operational implications of the tuned parameters in real systems merit discussion. In networked learning applications deployed in practice, parameters tuned via federated swarm optimization may need to satisfy operational constraints, such as limits on model size, energy consumption on edge devices, or fairness considerations across client groups. Integrating such constraints into the optimization formulation can be achieved via feasible sets and penalty terms, and experiments can explore how these constraints affect both the search process and the resulting models [62]. While the experiments outlined here are stylized, they can provide insights into how federated swarm optimization might behave in more complex, real-world deployments and help identify configurations and mechanisms that achieve a balanced compromise among accuracy, privacy, and resource usage.

7. Conclusion

This paper has examined federated swarm optimization as a method for privacy-preserving parameter tuning in networked learning systems. The analysis began with a formal model of distributed clients, each holding local data and loss functions, and introduced a shared parameter vector whose values influence learning performance across the network [63]. By distributing a swarm of candidate parameter configurations across clients and coordinating their evolution via a central server, the method seeks to explore the parameter space while respecting constraints on data movement and privacy.

The algorithmic framework adapts classical swarm dynamics by restricting direct access to global loss evaluations and by structuring communication around privacy-preserving summaries of local performance [64]. Clients maintain local best positions based on evaluations on their own data, while the server aggregates noisy or coarse reports to construct a reference position that guides collective behavior. The integration of noise and secure aggregation mechanisms aims to reduce the information available to potential adversaries, at the cost of introducing uncertainty into the optimization trajectory. Analytical considerations suggest that, under simplified conditions and moderate noise, the swarm can still converge toward regions with good global performance, although convergence may be to a neighborhood of an optimum rather than to an exact minimizer [65].

Complexity considerations indicate that the computational cost is dominated by local evaluations of candidate parameters, while communication cost depends strongly on the frequency and richness of reports from clients. Strategies such as event-driven reporting, position quantization, and limited numbers of transmitted candidates per round can reduce communication, with corresponding impacts on convergence speed and stability. Experimental designs outlined in the discussion section highlight how the framework can be studied empirically using synthetic or partitioned datasets, allowing examination of trade-offs among accuracy, communication, and privacy [66].

Federated swarm optimization provides a structured way to address parameter tuning in settings where data cannot be centralized and where privacy and communication constraints are significant. The framework accommodates heterogeneous clients and allows for integration of various privacy mechanisms and communication strategies. Future investigations can refine theoretical analyses, explore additional algorithmic variants, and examine behavior in more complex and realistic networked learning environments, including those with dynamic client populations, nonstationary data distributions, and multiobjective criteria that extend beyond predictive accuracy alone [67].

Acknowledgments

We would like to thank the reviewers of this research for their helpful comments and suggestions.

References

[1] A. Nejat, P. Mirzabeygi, and M. S. Panahi, "Airfoil shape optimization using improved multiobjective territorial particle swarm algorithm with the objective of improving stall characteristics," *Structural and Multidisciplinary Optimization*, vol. 49, pp. 953–967, 12 2013.

[2] X. Zhang and X. Zhang, "Shift based adaptive differential evolution for pid controller designs using swarm intelligence algorithm," *Cluster Computing*, vol. 20, pp. 291–299, 11 2016.

[3] D. Kumutha and N. A. Prabha, "Hybrid stbc-pts with enhanced artificial bee colony algorithm for papr reduction in mimo-ofdm system," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–17, 10 2017.

[4] R. Chandrasekar and S. Misra, "Using zonal agent distribution effectively for routing in mobile ad hoc networks," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 3, no. 2, pp. 82–89, 2008.

[5] S. Alyaseri and A. Aljanaby, "Population based heuristic approaches for grid job scheduling," *International Journal of Computer Applications*, vol. 91, pp. 45–50, 4 2014.

[6] M. Ding, H. Chen, N. Lin, S. Jing, F. Liu, X. Liang, and W. Liu, "Dynamic population artificial bee colony algorithm for multi-objective optimal power flow.," *Saudi journal of biological sciences*, vol. 24, pp. 703–710, 1 2017.

[7] A. Ławrynowicz, "Genetic algorithms for solving scheduling problems in manufacturing systems," *Foundations of Management*, vol. 3, pp. 7–26, 1 2011.

[8] C.-W. Tsai, C.-F. Lai, and A. V. Vasilakos, "Future internet of things: open issues and challenges," *Wireless Networks*, vol. 20, pp. 2201–2217, 5 2014.

[9] Z. J. Lim and M. W. Mustafa, "Evolved intelligent clustered bee colony for voltage stability prediction on power transmission system," *Soft Computing*, vol. 20, pp. 3215–3230, 5 2015.

[10] P. E. Rybski, A. Larson, H. Veeraraghavan, M. D. Anderson, and M. Gini, "Performance evaluation of a multi-robot search & retrieval system: Experiences with mindart," *Journal of Intelligent and Robotic Systems*, vol. 52, pp. 363–387, 5 2008.

[11] J. Beal, "Functional blueprints: an approach to modularity in grown systems," *Swarm Intelligence*, vol. 5, pp. 257–281, 6 2011.

[12] S. K. Rakesh, B. Choudhary, and R. Sandhu, "Cooperative problem solving in telecommunication network," *INTERNATIONAL JOURNAL OF MANAGEMENT & INFORMATION TECHNOLOGY*, vol. 4, pp. 388–392, 7 2013.

[13] H. Xue, X. Li, and H.-X. Ma, "Random fuzzy chance-constrained programming based on adaptive chaos quantum honey bee algorithm and robustness analysis," *International Journal of Automation and Computing*, vol. 7, pp. 115–122, 2 2010.

[14] R. Chandrasekar, V. Vijaykumar, and T. Srinivasan, "Probabilistic ant based clustering for distributed databases," in *2006 3rd International IEEE Conference Intelligent Systems*, pp. 538–545, IEEE, 2006.

[15] Y. Kang, H. Li, C. Wang, and L. Dai, "A hybrid big bang-big crunch algorithm for energy optimization on heterogeneous distributed system," *MATEC Web of Conferences*, vol. 119, pp. 01047–, 8 2017.

[16] K. A. Al-Anbarri and H. M. Naief, "Application of artificial bee colony algorithm in power flow studies," *UHD Journal of Science and Technology*, vol. 1, pp. 11–16, 4 2017.

[17] S. Banerjee and N. Agarwal, "Analyzing collective behavior from blogs using swarm intelligence," *Knowledge and Information Systems*, vol. 33, pp. 523–547, 6 2012.

[18] K. Li, C. E. Torres, K. Thomas, L. F. Rossi, and C.-C. Shen, "Slime mold inspired routing protocols for wireless sensor networks," *Swarm Intelligence*, vol. 5, pp. 183–223, 11 2011.

[19] S. Arjunan and P. Sujatha, "Lifetime maximization of wireless sensor network using fuzzy based unequal clustering and aco based routing hybrid protocol," *Applied Intelligence*, vol. 48, pp. 2229–2246, 11 2017.

[20] A. Kanakia, B. Touri, and N. Correll, "Modeling multi-robot task allocation with limited information as global game," *Swarm Intelligence*, vol. 10, pp. 147–160, 4 2016.

[21] F. S. Milani and A. H. Navin, "Multi-objective task scheduling in the cloud computing based on the patrice swarm optimization," *International Journal of Information Technology and Computer Science*, vol. 7, pp. 61–66, 4 2015.

[22] S. Keerthi, A. K, and M. Vijaykumar, "Survey paper on swarm intelligence," *International Journal of Computer Applications*, vol. 115, pp. 8–12, 4 2015.

[23] X. Li, L. Xu, H. Wang, J. Song, and S. X. Yang, "A differential evolution-based routing algorithm for environmental monitoring wireless sensor networks," *Sensors (Basel, Switzerland)*, vol. 10, pp. 5425–5442, 6 2010.

[24] V. Vijaykumar, R. Chandrasekar, and T. Srinivasan, "An obstacle avoidance strategy to ant colony optimization algorithm for classification in event logs," in *2006 IEEE Conference on Cybernetics and Intelligent Systems*, pp. 1–6, 2006.

[25] M. M. al Rifaie, F. F. Leymarie, W. Latham, and J. M. Bishop, "Swarmic autopoiesis and computational creativity," *Connection Science*, vol. 29, pp. 276–294, 1 2017.

[26] S. Molaiy and M. Effatparvar, "Scheduling in grid systems using ant colony algorithm," *International Journal of Computer Network and Information Security*, vol. 6, pp. 19–22, 1 2014.

[27] S. P. Fekete, A. W. Richa, K. Römer, and C. Scheideler, "Algorithmic foundations of programmable matter dagstuhl seminar 16271," *ACM SIGACT News*, vol. 48, pp. 87–94, 6 2017.

[28] P. Rabanal, I. Rodríguez, and F. Rubio, "An aco-rfd hybrid method to solve np-complete problems," *Frontiers of Computer Science*, vol. 7, pp. 729–744, 10 2013.

[29] A. J. Trewavas and F. Baluška, "The ubiquity of consciousness," *EMBO reports*, vol. 12, pp. 1221–1225, 12 2011.

[30] C. Yinka-Banjo, I. O. Osunmakinde, and A. Bagula, "Cooperative behaviours with swarm intelligence in multi-robot systems for safety inspections in underground terrains," *Mathematical Problems in Engineering*, vol. 2014, pp. 1–20, 7 2014.

[31] K. K. Bhattacharjee and S. P. Sarmah, "Modified swarm intelligence based techniques for the knapsack problem," *Applied Intelligence*, vol. 46, pp. 158–179, 8 2016.

[32] B. P. De, R. Kar, D. Mandal, and S. P. Ghoshal, "Pso with aging leader and challengers for optimal design of high speed symmetric switching cmos inverter," *International Journal of Machine Learning and Cybernetics*, vol. 8, pp. 1403–1422, 3 2016.

[33] M. Sengaliappan and K. Kumaravel, "Performance study on multipath routing algorithm using temporarily ordered routing algorithm (tora) using link – reversal in wireless networks," *International Journal of Trend in Scientific Research and Development*, vol. Volume-2, pp. 328–331, 12 2017.

[34] R. Chandrasekar and T. Srinivasan, "An improved probabilistic ant based clustering for distributed databases," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI*, pp. 2701–2706, 2007.

[35] M. E. Valentinuzzi, "Handbook of bioinspired algorithms and applications - handbook of bioinspired algorithms and applications," *BioMedical Engineering OnLine*, vol. 5, pp. 47–47, 7 2006.

[36] Y. Zeng, E. Zhou, Y. Wang, X. Ren, Y. Qin, Z. Huang, and N. Zhong, "Research interests: their dynamics, structures and applications in unifying search and reasoning," *Journal of Intelligent Information Systems*, vol. 37, pp. 65–88, 12 2010.

[37] A. Campo, Álvaro Gutiérrez, S. Nouyan, C. Pinciroli, V. Longchamp, S. Garnier, and M. Dorigo, "Artificial pheromone for path selection by a foraging swarm of robots," *Biological cybernetics*, vol. 103, pp. 339–352, 7 2010.

[38] L. Golshanara, S. M. R. Rankoohi, and H. Shah-Hosseini, "A multi-colony ant algorithm for optimizing join queries in distributed database systems," *Knowledge and Information Systems*, vol. 39, pp. 175–206, 1 2013.

[39] L. Sun, L. Wang, and H. Ge, "A coevolutionary bacterial foraging model using pso in job-shop scheduling environments," *International Journal of Grid and Distributed Computing*, vol. 9, pp. 379–394, 9 2016.

[40] G. Raghavendra and M. Ramachandra, "Situational analysis of distributed system and its effectiveness in area of power system," *International Journal of Computer Applications*, vol. 103, pp. 23–30, 10 2014.

[41] I. N. Dolgopolov, "Modeling of a multifactor therapeutic effect on a biosystem," *Cybernetics and Systems Analysis*, vol. 48, pp. 660–672, 10 2012.

[42] S. S. Yadav and A. Chitra, "Tbft: An energy efficient modelling of wsn using tree-based fusion technique," *Wireless Personal Communications*, vol. 97, pp. 1217–1234, 6 2017.

[43] R. Rao and V. Patel, "Comparative performance of an elitist teaching-learning-based optimization algorithm for solving unconstrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 4, pp. 29–50, 1 2013.

[44] V. Vijaykumar, R. Chandrasekar, and T. Srinivasan, "An ant odor analysis approach to the ant colony optimization algorithm for data-aggregation in wireless sensor networks," in *2006 International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1–4, IEEE, 2006.

[45] M.-H. Lim, Y.-S. Ong, and S. Gustafson, "Editorial," *Memetic Computing*, vol. 7, pp. 157–158, 8 2015.

[46] B. Samanta and C. Nataraj, "Application of particle swarm optimization and proximal support vector machines for fault detection," *Swarm Intelligence*, vol. 3, pp. 303–325, 5 2009.

[47] Z. Zhang, W. Huangfu, K. Long, X. Zhang, L. Xiaoyuan, and B. Zhong, "On the designing principles and optimization approaches of bio-inspired self-organized network: a survey," *Science China Information Sciences*, vol. 56, pp. 1–28, 7 2013.

[48] P. Gowtham and P. V. Karthick, "Optimized coverage and efficient load balancing algorithm for wsns-a survey," *International Journal of Engineering Research and Science*, vol. 3, pp. 90–94, 5 2017.

[49] M. Ciszak, D. Comparini, B. Mazzolai, F. Baluška, F. T. Arecchi, T. Vicsek, and S. Mancuso, "Swarming behavior in plant roots.," *PloS one*, vol. 7, pp. e29759–, 1 2012.

[50] M. K. M. Zamani, I. Musirin, S. I. Suliman, M. M. Othman, and M. F. M. Kamal, "Multi-area economic dispatch performance using swarm intelligence technique considering voltage stability," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 7, pp. 1–7, 2 2017.

[51] H. Su, "Siting and sizing of distributed generators based on improved simulated annealing particle swarm optimization.," *Environmental science and pollution research international*, vol. 26, pp. 17927–17938, 12 2017.

[52] H. Liu, F. Sun, and S. Wang, "Virtual strategy qos routing in satellite networks," *Science China Information Sciences*, vol. 59, pp. 92201–, 8 2016.

[53] A. Prakasam and N. Savarimuthu, "Metaheuristic algorithms and probabilistic behaviour: a comprehensive analysis of ant colony optimization and its variants," *Artificial Intelligence Review*, vol. 45, pp. 97–130, 10 2015.

[54] C. Ramachandran, S. Misra, and M. Obaidat, "On evaluating some agent-based intrusion detection schemes in mobile ad-hoc networks," in *Proceedings of the SPECTS 2007*, (San Diego, CA), pp. 594–601, July 2007.

[55] R. Mardian and K. Sekiyama, "Ant systems-based dna circuits," *BioNanoScience*, vol. 5, pp. 206–216, 11 2015.

[56] C. Cecati and P. Siano, "Special issue on advanced computational intelligence systems for smart grids planning and management," *Journal of Ambient Intelligence and Humanized Computing*, vol. 4, pp. 603–604, 8 2013.

[57] M. A. Abd-Allah, A. Said, and M. N. Ali, "Mitigation of lightning hazards at the more sensitive points in wind farms using ant-colony optimization technique," *Bulletin of Electrical Engineering and Informatics*, vol. 5, 6 2016.

[58] S. G. Santhi and K. Venkatachalapathy, "Multiple cluster tree routing and scheduling for collision avoidance in 802.15.4. sensor networks," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 7, pp. 3075–3082, 4 2014.

[59] V. Mangat, "Swarm intelligence based technique for rule mining in the medical domain," *International Journal of Computer Applications*, vol. 4, pp. 19–24, 7 2010.

[60] M. N. Abdurrazaq, B. R. Trilaksono, and B. Rahardjo, "Dids using cooperative agents based on ant colony clustering," *Journal of ICT Research and Applications*, vol. 8, pp. 213–233, 8 2015.

[61] A. H. Al-Mter and S. Lu, "A p article swarm optimization algorithm based on uniform design," *International Journal of Data Mining & Knowledge Management Process*, vol. 6, pp. 29–35, 3 2016.

[62] X. Guo, T. Hu, C. Wu, T. Zhang, and Y. Lv, "Multiobjective optimization of the proposed multi-reservoir operating policy using improved nspso," *Water Resources Management*, vol. 27, pp. 2137–2153, 2 2013.

[63] M. A. Z. Raja, J. A. Khan, S. ul Islam Ahmad, and I. M. Qureshi, "A new stochastic technique for painlevé equation-i using neural network optimized with swarm intelligence," *Computational intelligence and neuroscience*, vol. 2012, pp. 4–10, 7 2012.

[64] C. Ramachandran, R. Malik, X. Jin, J. Gao, K. Nahrstedt, and J. Han, "Videomule: a consensus learning approach to multi-label classification from noisy user-generated videos," in *Proceedings of the 17th ACM international conference on Multimedia*, pp. 721–724, 2009.

[65] E. Tuğcu, I. Kaya, and A. Yazgan, "Cmf-dfe based adaptive blind equalization using particle swarm optimization," *Radioengineering*, vol. 25, pp. 124–131, 4 2016.

[66] M. Abd-Allah, A. Said, and M. N. Ali, "Mitigation of lightning hazards at the more sensitive points in wind farms using ant-colony optimization technique," *Bulletin of Electrical Engineering and Informatics*, vol. 5, pp. 144–159, 6 2016.

[67] S. O. Palermos, "The dynamics of group cognition," *Minds and Machines*, vol. 26, pp. 409–440, 10 2016.