Consensus-ADMM-Integrated Particle Swarm Optimization for Distributed Convex–Nonconvex Programs Over Time-Varying Graphs

Rizky A. Pratama[1] **and Siti Nurhaliza Putri**[2]

[1] Nusantara Institute of Technology, Department of Computer Science and Intelligent Systems, Jalan Diponegoro 112, Yogyakarta 55233, Indonesia
[2] Bandung Raya University, Faculty of Computing and Digital Media, Jalan Merdeka 45, Bandung 40115, Indonesia

ABSTRACT Distributed optimization over multi-agent networks arises in estimation, control, and learning tasks where data is geographically dispersed and communication is constrained. Classical distributed convex optimization methods rely on strong convexity and static graph assumptions, while many emerging applications feature nonconvex local objectives and time-varying communication topologies. Metaheuristic search strategies, such as particle swarm optimization, have been used to explore complex landscapes, but they often lack a systematic treatment of consensus and dual variables. This work considers a distributed composite convex–nonconvex program defined over a network of agents connected by a time-varying graph. Each agent possesses a local cost function with both convex and nonconvex components and participates in a consensus constraint enforced through local interactions. A consensus-based alternating direction method of multipliers is integrated with a particle swarm mechanism in order to combine structured augmented Lagrangian updates with stochastic exploration in the primal domain. The resulting scheme allows each agent to maintain local primal, dual, and velocity states while exchanging low-dimensional consensus variables with neighbors. The algorithm is described in a unified operator form that highlights the interaction between consensus steps, dual ascent, and swarm-based perturbations. Conditions on the communication weights, penalty parameters, and search coefficients are discussed under which consensus is asymptotically preserved and the network trajectories approach stationary solutions of the nonconvex problem. Implementation aspects for large networks and numerical illustration scenarios are outlined to indicate how the method behaves in representative distributed optimization tasks with heterogeneous local models.

KEYWORDS

1. Introduction

Distributed optimization techniques have become an essential tool for handling data and decision variables that are scattered across multiple computational entities [1]. In many applications, such entities represent physical agents, sensing devices, or computing nodes that can only communicate over a constrained network. The joint behavior of the network is then characterized by a set of local objective functions and constraints that must be treated in a coordinated way. Classical formulations often assume convexity of the local objectives and rely on the presence of a static and connected communication graph. In contrast, several modern applications lead to nonconvex objective components and time-varying communication patterns [2]. Examples include distributed training of nonlinear models, nonconvex regularization for high-dimensional estimation, and control problems involving nonlinear dynamics with intermittent communication.

The alternating direction method of multipliers has been widely used as a flexible framework for decomposing large optimization problems. In its consensus form, it introduces local copies of a global decision vector and enforces agreement through augmented Lagrangian terms and dual variables. When

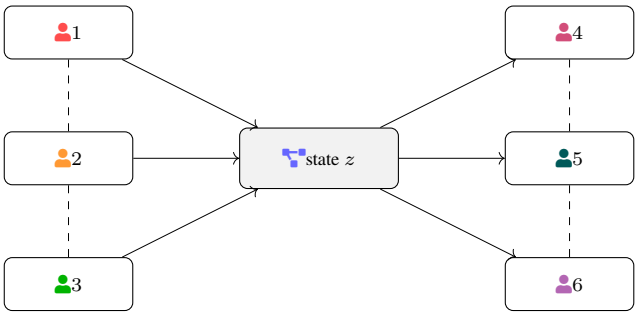

Figure 1 Network-level view of agents maintaining local variables while interacting with a virtual consensus state. Solid arrows depict augmented Lagrangian coupling, and dashed links indicate peer-to-peer communication in the time-varying graph.

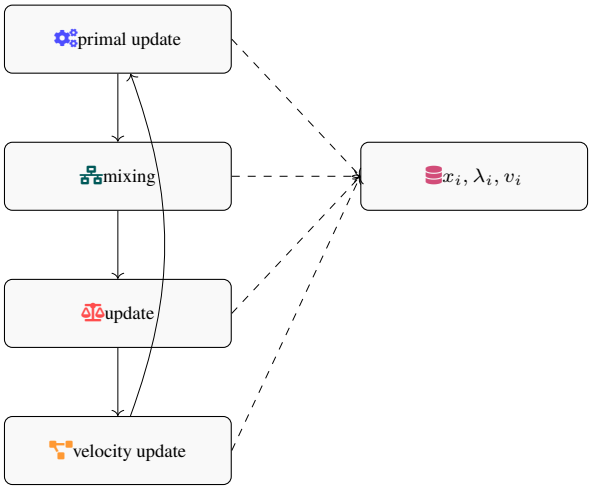

Figure 2 Algorithmic pipeline at one agent combining augmented Lagrangian updates, consensus operations, dual ascent, and swarm dynamics, with a compact local state shared across the stages.

applied in a distributed environment, consensus-based ADMM decomposes the optimization task into local subproblems that can be solved in parallel by the agents, with coordination achieved via exchange of intermediate variables. This structure is suitable for convex programs and can tolerate moderately complex communication patterns [3]. However, in the presence of nonconvex objectives, classical convergence guarantees no longer apply directly, and additional mechanisms are often required to handle local minima, saddle points, and flat regions.

Particle swarm optimization provides a heuristic approach for exploring complex nonconvex landscapes. It represents the search space through a set of particles, each characterized by a position and velocity that are updated iteratively. The update rules combine inertia with local and global attraction terms that guide the swarm toward promising regions. In centralized implementations, PSO can be applied to nonconvex problems without direct use of gradient information, making it attractive when derivatives are expensive or unavailable. Nonetheless, vanilla PSO does not incorporate explicit consensus constraints and does not naturally account for the network structure that arises in distributed optimization.

The integration of consensus-based ADMM with particle swarm mechanisms is motivated by the desire to balance structured augmented Lagrangian updates with exploratory search. The augmented Lagrangian framework provides a principled way to enforce consensus and linear constraints, while PSO-type dynamics introduce exploration and diversity in the primal iterates. In a distributed convex–nonconvex setting over time-varying graphs, such a combination allows each agent to exploit local problem structure while remaining sensitive to global swarm information propagated through the network.

Time-varying communication graphs arise when links between agents are affected by mobility, fading, scheduling policies, or external disruptions [4]. A network sequence that is not connected at every iteration may still exhibit joint connectivity over time windows. For distributed optimization algorithms, this raises the challenge of maintaining consensus in the presence of intermittent connectivity and nonstationary mixing patterns. Weight matrices that satisfy standard stochasticity and connectivity properties can still be used to propagate information, but convergence analysis must account for the temporal variability.

In this work, a distributed convex–nonconvex optimization problem is considered in which each agent holds a composite local objective with both convex and nonconvex parts. A consensus-based ADMM scheme is integrated with a particle swarm component that acts on the local primal variables [5]. The integration is designed so that the core ADMM structure, including augmented Lagrangian minimization and dual ascent, is preserved, while the PSO velocities introduce controlled perturbations that enhance exploration of the nonconvex landscape.

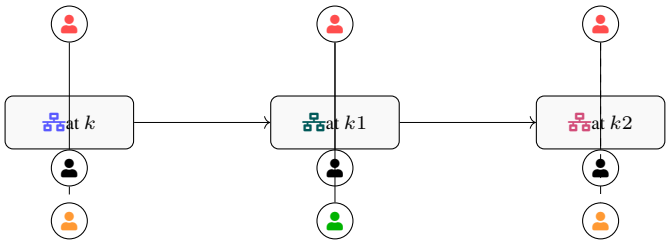

Figure 3 Snapshot sequence of a time-varying communication topology, illustrating how local neighborhoods change across iterations while preserving joint connectivity over time windows.

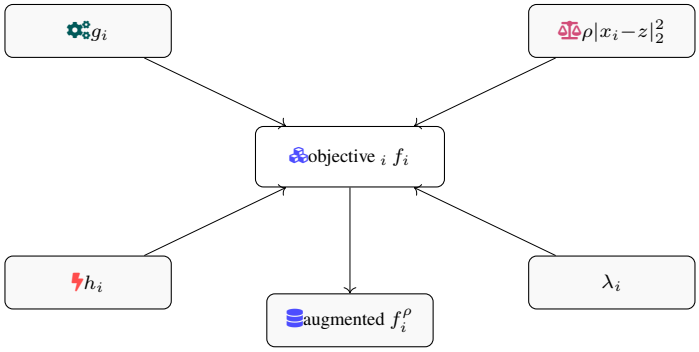

Figure 4 Structural decomposition of the composite objective at each agent, including convex and nonconvex components, quadratic penalties, and dual contributions forming the augmented local cost.

The communication among agents is modeled by a time-varying graph, and the consensus steps incorporate a sequence of mixing matrices that satisfy standard connectivity and regularity conditions.

The subsequent sections introduce the network and problem setting, develop the integrated consensus–ADMM–PSO algorithm, examine its convergence properties under suitable assumptions, and discuss implementation considerations for large-scale networks. Numerical scenarios are outlined to illustrate how the method operates when applied to representative distributed optimization tasks [6]. The overall exposition emphasizes the interplay between consensus enforcement, dual variable evolution, and swarm-based search in a distributed nonconvex context [7].

2. Preliminaries on Distributed Optimization and Time-Varying Graphs

Consider a collection of agents indexed by the set

$$\mathcal{V} = \{1, 2, \ldots, N\}.$$

Each agent i maintains a local copy of a global decision vector in a Euclidean space. The agents exchange information over a sequence of undirected graphs that evolve over discrete time. At iteration k, the active communication pattern is described by a graph [8]

$$\mathcal{G}^k = \mathcal{V}, \mathcal{E}^k,$$

where $\mathcal{E}^k$ is the set of edges representing the links available at that iteration. If $i, j \in \mathcal{E}^k$, agents i and j can exchange messages at iteration k. The sequence $\{\mathcal{G}^k\}_{k\geq 0}$ is allowed to vary arbitrarily, subject to connectivity assumptions that hold over finite windows.

The communication process is modeled through a sequence of weight matrices. At each iteration k, there is a matrix

$$W^k \in \mathbb{R}^{N\times N},$$

where the entry w_{ij}^k quantifies the influence of agent j on agent i at that iteration. It is assumed that $w_{ij}^k > 0$ if and only if $i = j$ or $i, j \in \mathcal{E}^k$. In addition, the matrix W^k respects standard stochasticity conditions, such as row stochasticity

$$\sum_{j=1}^{N} w_{ij}^k = 1 \quad \text{for all } i \text{ and } k.$$

In some cases, column stochasticity or doubly stochasticity is also imposed, but it is not strictly required for the formulation.

The distributed optimization problem considered in this setting involves a global decision vector $x \in \mathbb{R}^d$ shared across the network. Each agent i has access only to a local objective function

$$f_i : \mathbb{R}^d \to \mathbb{R},$$

and the global optimization task is expressed as [9]

$$\min_{x\in\mathbb{R}^d} \sum_{i=1}^{N} f_i x.$$

In convex settings, each function f_i is assumed to be convex, and various distributed gradient and consensus-based methods can be applied. In the present context, the local functions are composite, with a convex component and a nonconvex component. This introduces additional challenges in the analysis and calls for mechanisms that can explore nonconvex regions.

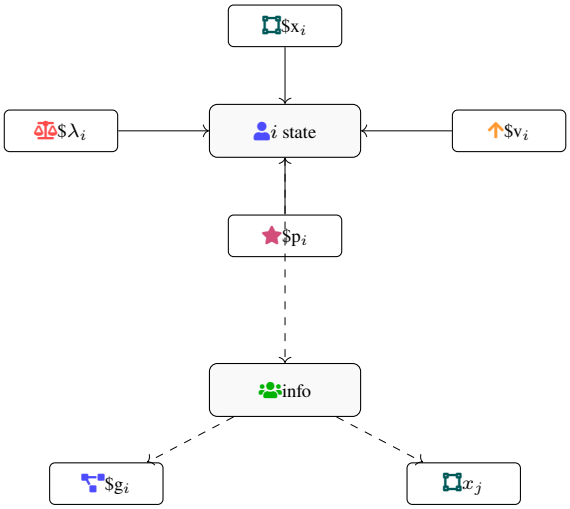

Figure 5 Local storage layout for one agent, showing primal, dual, velocity, and personal-best variables, together with a neighborhood structure holding aggregated information used by the swarm and consensus mechanisms.

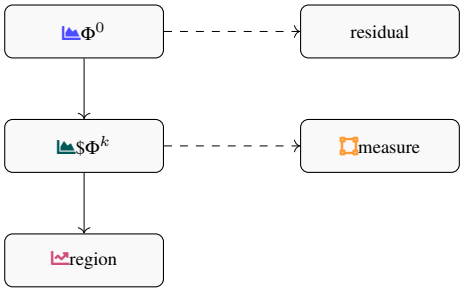

Figure 6 Conceptual trajectory of a Lyapunov functional aggregating objective values, dual norms, and disagreement terms, together with associated residual measures decreasing toward a stationary regime.

Consensus-based methods approach the global problem by introducing local copies of the decision vector [10]. Each agent holds a variable

$$x_i \in \mathbb{R}^d,$$

and consensus is enforced by requiring that all local variables coincide. The problem can then be written in the form

$$\min_{x_1,\dots,x_N} \sum_{i=1}^{N} f_i x_i,$$

subject to [11]

$$x_i = x_j \quad \text{for all } i,j \in \mathcal{E}^k \quad \text{and for all } k.$$

In this representation, the constraints are time-varying, reflecting the changing graph structure. A more convenient formulation introduces a vector of stacked local variables

$$\mathbf{x} = \left(x_1^\top, \dots, 12x_N^\top\right)^\top \in \mathbb{R}^{Nd},$$

together with linear operators that encode the consensus constraints.

Classical consensus algorithms rely on repeated multiplication by the weight matrices. For example, a sequence of iterates satisfying

$$x_i^{k1} = \sum_{j=1}^{N} w_{ij}^k x_j^k$$

can converge to a common value under appropriate conditions. For static graphs, such conditions include connectivity and suitable spectral properties of the matrix [13]. For time-varying graphs, joint connectivity over time windows and uniform spectral bounds are typically imposed. These conditions ensure that the influence of any agent eventually propagates through the network.

The alternating direction method of multipliers is formulated by splitting the problem into blocks and augmenting the Lagrangian with quadratic penalty terms. In a consensus setting, one often introduces a global variable z that represents the common value of the local variables and enforces equality constraints $x_i = z$ through dual variables. The augmented Lagrangian associated with this formulation takes the form of a sum of local terms plus consensus penalties [14]. The ADMM iterations then alternate between minimizing with respect to the primal variables and updating the dual variables.

Particle swarm optimization introduces an additional layer of dynamics to the primal iterates. Each agent maintains position and velocity vectors, often denoted by

$$x_i^k \quad \text{and} \quad v_i^k,$$

respectively. The velocity is updated using an inertia term, a cognitive term that attracts the particle toward its personal best position, and a social term that attracts it toward a global or

Component	Symbol	Space	Role
Local primal state	x_i	$\mathbb{R}^d$	decision vector
Dual variable	λ_i	$\mathbb{R}^d$	consensus tension
Velocity	v_i	$\mathbb{R}^d$	swarm motion
Personal best	p_i	$\mathbb{R}^d$	local archive
Consensus proxy	y_i	$\mathbb{R}^d$	mixed neighbor state
Global reference (avg.)	$\bar{x}$	$\mathbb{R}^d$	network mean

Table 1 Core per-agent state variables and their roles in the combined consensus–ADMM–PSO dynamics.

Parameter	Nominal	Range	Effect
Penalty ρ	10^{-1}	$10^{-3}, 10^{1}$	consensus strength
Step α^k	decaying	$0, 10^{-1}$	stability vs speed
Inertia ω	0.6	$0.3, 0.9$	exploration depth
Cognitive c_1	1.2	$0.5, 2$	bias to p_i
Social c_2	1.2	$0.5, 2$	bias to neighborhood best
Velocity clipping	yes	bounded norm	limits oscillations

Table 2 Typical parameter magnitudes and qualitative influence on the distributed dynamics.

neighborhood best position [15]. Random coefficients modulate the influence of these terms at every iteration. The position is then updated by adding the new velocity.

In a distributed setting with time-varying graphs, the social term of PSO can be localized so that each agent is attracted toward the best positions observed within its neighborhood. This leads to a structure in which the communication graph influences the swarm dynamics in a similar way as in consensus algorithms. When such dynamics are combined with ADMM steps, the resulting scheme must carefully balance the deterministic augmented Lagrangian updates with the stochastic exploration induced by the swarm [16].

The present work uses the above elements to build an integrated algorithm where each agent simultaneously engages in consensus, dual ascent, and PSO-based exploration. The time-varying graph provides the underlying communication pattern, the weight matrices define the consensus mixing, and the swarm dynamics act as a perturbation of the ADMM primal updates.

3. Problem Formulation and Consensus–ADMM–PSO Scheme

The global optimization problem addressed in this work is formulated as follows. There is a global decision variable [17]

$$x \in \mathbb{R}^d,$$

and each agent i possesses a local cost function of the form

$$f_i x = g_i x \;\; h_i x,$$

where g_i is convex and h_i may be nonconvex [18]. The function g_i can represent a loss derived from local data, such as a convex empirical risk, while h_i can capture regularization or nonconvex penalties. The global optimization task is to minimize the aggregate objective

$$\min_{x \in \mathbb{R}^d} \; \sum_{i=1}^{N} \big(g_i x \;\; h_i x 19 \big),$$

possibly subject to additional linear constraints that can be incorporated into the local functions or into the ADMM structure.

To employ a consensus-based approach, a local copy

$$x_i \in \mathbb{R}^d$$

is associated with each agent i. The global optimization problem is cast as [20]

$$\min_{x_1,\ldots,x_N} \; \sum_{i=1}^{N} \big(g_i x_i \;\; h_i x_i \big),$$

subject to the consensus constraints [21]

$$x_i = x_j \quad \text{for all } i, j \in \mathcal{E}^k \quad \text{for all } k.$$

Equivalently, global consensus can be expressed by requiring

$$x_1 = x_2 = \cdots = x_N,$$

but the enforcement is carried out through local interactions on the time-varying graph. For the purposes of introducing ADMM, an auxiliary global variable [22]

$$z \in \mathbb{R}^d$$

is introduced. The problem is rewritten as

$$\min_{x_1,\ldots,x_N,z} \; \sum_{i=1}^{N} \big(g_i x_i \;\; h_i x_i 23 \big),$$

subject to

$$x_i = z \quad \text{for all } i.$$

The time-varying graph structure is incorporated into how the variable z is approximated through local consensus rather than as an explicit global variable. However, the above formulation is useful for defining the augmented Lagrangian.

For each agent i, a dual variable [24]

$$\lambda_i \in \mathbb{R}^d$$

is associated with the constraint $x_i = z$. The augmented Lagrangian can be expressed as

$$\mathcal{L}_\rho \big(x_1, \ldots, x_N, z, 25 \lambda_1, \ldots, \lambda_N \big) = \sum_{i=1}^{N} \big[g_i x_i \;\; h_i x_i 26 \;\; \lambda_i^\top x_i - z \;\; \frac{\rho}{2} \; \| x_i -$$

Graph property	Condition	Symbolic form	Role
Joint connectivity	window size Q finite	$\bigcup_{t=k}^{kQ-1} \mathcal{G}^t$ connected	information spread
Weight lower bound	$\eta > 0$	$w_{ij}^k \geq \eta$ on edges	avoids vanishing links
Row stochasticity	$_j\, w_{ij}^k = 1$	$W^k \mathbf{1} = \mathbf{1}$	mass preservation
Sparsity control	bounded degrees	$\lvert\mathcal{N}_i^k\rvert \leq d_{\max}$	communication scaling
Temporal variability	arbitrary under above	$k \mapsto \mathcal{G}^k$ free	models switching links

Table 3 Structural assumptions on the time-varying communication graph and weight sequence.

Step	Per-iteration cost	Per-agent memory	Communication
Local gradient / prox	$\mathcal{O}d$ to $\mathcal{O}d^2$	Od for buffers	none
Velocity update	$\mathcal{O}d$	Od (store v_i, p_i)	none
Consensus mixing	$\mathcal{O}d\lvert\mathcal{N}_i^k\rvert$	Od temporary sum	send x_i to neighbors
Dual update	$\mathcal{O}d$	Od (store λ_i)	none
Neighborhood best	$\mathcal{O}\lvert\mathcal{N}_i^k\rvert$	small scalars/vectors	exchange scores or p_j

Table 4 Sketch of per-agent computational, storage, and communication complexity per iteration.

where $\rho > 0$ is a penalty parameter [27]. Classical ADMM alternates between minimizing the augmented Lagrangian with respect to the primal variables and updating the dual variables.

In the integrated consensus–ADMM–PSO scheme, each agent maintains the triplet of local states

$$x_i^k, \quad \lambda_i^k, \quad v_i^k,$$

where x_i^k is the current position, λ_i^k is the current dual variable, and v_i^k is the PSO velocity. The algorithm proceeds in iterations indexed by k. At each iteration, three types of updates occur at each agent: local primal update influenced by the augmented Lagrangian and the velocity; consensus-based mixing of primal or auxiliary variables; and dual variable update.

The local primal update at agent i seeks to approximately minimize a local augmented Lagrangian term [28]. A typical update has the form

$$x_i^{k\frac{1}{2}} = \arg\min_{x_i} \; g_i x_i \;\; h_i x_i \\ \lambda_i^{k\top} x_i - z^k \;\; \frac{\rho}{2} \lVert x_i - z^k \rVert_2^2,$$

where z^k is an approximation of the consensus variable obtained from the previous iteration. This minimization may not be carried out exactly, especially in the presence of nonconvexity [29]. Instead, a proximal or gradient-based step can be used to approximate the minimizer. For example, a linearization of the nonconvex component can be introduced, leading to an update of the form

$$x_i^{k\frac{1}{2}} = x_i^k - \alpha^k \left(\nabla g_i x_i^k \;\; \tilde{\nabla} h_i x_i^k \;\; \lambda_i^k \;\; \rho x_i^k - z^k \right),$$

where α^k is a step size and $\tilde{\nabla} h_i$ denotes a generalized gradient or approximate gradient of the nonconvex term.

The particle swarm component introduces a velocity update for each agent [30]. Each agent keeps track of a personal best position

$$p_i^k,$$

representing the best value of its local objective encountered up to iteration k, and also has access to a neighborhood best position

$$g_i^k,$$

which is obtained by aggregating information from neighboring agents. The velocity update follows the standard structure

$$v_i^{k1} = \omega 31 v_i^k \;\; c_1 r_1^k \left(p_i^k - x_i^k \right) \\ c_2 r_2^k \left(32 g_i^k - x_i^k \right),$$

where ω is the inertia coefficient, c_1 and c_2 are cognitive and social coefficients, and r_1^k and r_2^k are independent random variables uniformly distributed in the interval $0, 1$. The intermediate primal iterate is then modified by the velocity through

$$x_i^{k1} = x_i^{k\frac{1}{2}} \;\; v_i^{k1}.$$

In this way, the ADMM-inspired local update is perturbed by a PSO velocity that reflects local and neighborhood historical information.

The consensus mechanism uses the time-varying weight matrices. Each agent computes a local consensus proxy [33]

$$y_i^{k1} = \sum_{j=1}^{N} w_{ij}^k x_j^{k1},$$

which aggregates the post-velocity positions of its neighbors, including itself. The variable y_i^{k1} acts as a local approximation to the global variable z^{k1}. Depending on the chosen formulation, either z^{k1} is explicitly computed as an average of the y_i^{k1}, or the local proxies y_i^{k1} are used directly in the dual updates. A simple choice is to identify

$$z^{k1} = \frac{1}{N} \sum_{i=1}^{N} x_i^{k1},$$

which can be approximated through repeated consensus steps using the matrices W^k.

The dual update is carried out locally at each agent according to

$$\lambda_i^{k1} = \lambda_i^k \;\; \rho \left(x_i^{k1} - z^{k1} \right) 34.$$

When z^{k1} is replaced by a consensus proxy y_i^{k1}, the update becomes

$$\lambda_i^{k1} = \lambda_i^k \;\; \rho \left(x_i^{k1} - y_i^{k1} \right).$$

This update can be viewed as a local dual ascent step that tries to enforce agreement between the primal variable and its local consensus approximation.

Setting	ω	c_1	c_2	Behavior
Exploratory swarm	0.8	1.5	1.5	strong exploration
Balanced search	0.6	1.2	1.2	trade-off regime
Conservative refinement	0.4	0.8	0.8	local polishing
Neighborhood-focused	0.6	0.7	1.6	more social pull
Self-focused	0.6	1.6	0.7	more personal pull

Table 5 Representative PSO coefficient combinations shaping the exploration intensity in the integrated method.

Quantity	Prototype form	Scale	Use
Consensus gap	$\frac{1}{N}\ _i\ x_i - \bar{x}\ _2^2$	state level	agreement tracking
Primal residual	$\ _i\ x_i - z\ _2^2$	constraint level	penalty tuning
Dual norm	$\ _i\ \lambda_i\ _2^2$	dual level	multiplier growth
Lyapunov proxy	$\mathbf{\Phi}^k$ aggregate	mixed scale	qualitative descent
Velocity energy	$\ _i\ v_i\ _2^2$	swarm level	exploration monitoring

Table 6 Typical scalar indicators used to monitor convergence, consensus, and swarm activity.

The integrated consensus–ADMM–PSO scheme thus couples the augmented Lagrangian structure with PSO dynamics and consensus mixing. The presence of nonconvex terms in the local objectives is handled by a combination of proximal approximations and stochastic exploration through the velocities [35]. The time-varying graphs are reflected in the sequence of weight matrices used in the consensus proxies and neighborhood best computations.

4. Convergence Properties and Complexity Analysis

The convergence analysis of the consensus–ADMM–PSO scheme considers the interplay between three interacting mechanisms: the augmented Lagrangian minimization, the dual ascent, and the particle swarm exploration. The objective is to identify conditions under which the sequence of iterates remains bounded, consensus is asymptotically achieved, and limit points satisfy stationary conditions for the nonconvex optimization problem.

Assume that each convex component g_i is proper, lower semicontinuous, and has Lipschitz continuous gradient on its effective domain. The nonconvex component h_i is assumed to be differentiable with a locally Lipschitz gradient or, more generally, to admit a subgradient satisfying suitable regularity properties [36]. It is further assumed that the aggregate objective is bounded below on $\mathbb{R}^d$, so that the optimization problem is well posed. The penalty parameter ρ is taken to be positive and fixed, although adaptive strategies can also be considered.

The time-varying graph sequence is assumed to satisfy a joint connectivity condition. There exists an integer $Q \geq 1$ such that for any integer k, the union of the graphs

$$\mathcal{G}^k, \mathcal{G}^{k1}, \ldots, 37\mathcal{G}^{kQ-1}$$

is connected. The associated sequence of weight matrices $\{W^k\}$ is assumed to be uniformly stochastic. Specifically, each W^k has nonnegative entries, satisfies row stochasticity, and there exists a positive constant η such that

$$w_{ij}^k \geq \eta$$

whenever i, j is an edge of $\mathcal{G}^k$ or $i = j$. These conditions ensure that the consensus dynamics induced by the matrices W^k converge to a common value in the absence of noise.

In addition to the consensus assumptions, control of the PSO velocities is required. The inertia weight ω and the coefficients c_1 and c_2 must be chosen so that the velocity process does not diverge. A common condition is that the spectral radius of the linearized velocity update is less than one in expectation. More concretely, if the random variables r_1^k and r_2^k have mean one half, then it is typical to require

$$\omega 38\ \frac{c_1}{2}\ \frac{c_2}{2} < 1.$$

This inequality is sufficient to guarantee that, in a linearized model, the velocities remain bounded and eventually decay. In the integrated scheme, further damping can be introduced by projecting the velocities onto a bounded set.

The augmented Lagrangian associated with the consensus formulation can be used to construct a Lyapunov function that captures the evolution of the network state. Consider the function

$$\mathbf{\Phi}^k = \ _{i=1}^N \left[39 g_i x_i^k\ \ h_i x_i^k\right]\ \frac{1}{2\rho}\ _{i=1}^N \|\lambda_i^k\|_2^2\ \frac{\gamma}{2}\ _{i=1}^N \|x_i^k - \bar{x}^k\|_2^2,$$

where $\bar{x}^k$ denotes the average of the local positions

$$\bar{x}^k = \frac{1}{N}\ _{i=1}^N x_i^k,$$

and γ is a positive constant. The first two terms reflect the local objective values and dual norms, while the last term measures the consensus disagreement. Under suitable assumptions on the step sizes and the velocity damping, it can be shown that the expected value of $\mathbf{\Phi}^k$ decreases along the iterations, modulo small perturbations induced by the stochastic terms.

The presence of PSO velocities means that the update for x_i^{k1} can be decomposed into a deterministic component coming from the augmented Lagrangian step and a stochastic component coming from the velocity. Denote the deterministic update by

$$\tilde{x}_i^{k1}$$

and write [40]

$$x_i^{k1} = \tilde{x}_i^{k1}\ \ v_i^{k1}.$$

Phase	Iteration range	Dominant effect	Qualitative trend
Initialization	early k	random velocities	rapid dispersion
Alignment	intermediate	consensus penalties	shrinking gaps
Exploration	intermediate	swarm dynamics	objective dips, spikes
Refinement	larger k	ADMM structure	smoother decrease
Stagnation band	late k	small updates	fluctuations near steady region

Table 7 High-level phases in the empirical temporal evolution of the distributed dynamics.

Scenario	Local model	Graph pattern	Objective shape	Swarm role
Nonlinear learning	logistic + nonconvex reg.	switching ring	multi-well	avoid shallow minima
Resource allocation	convex cost + penalties	clustered	partly flat	explore trade-offs
Matrix factorization	bilinear factors	random geometric	highly nonconvex	search multiple modes
Control tuning	quadratic + barriers	slowly varying mesh	nearly convex	fine-tune parameters
Sensing fusion	convex loss + concave reg	gossip-like	sparse-friendly	diversify supports

Table 8 Representative application types compatible with the consensus–ADMM–PSO formulation and its communication constraints.

The consensus term in the Lyapunov function can thus be expressed as

$$x_i^{k1} - \bar{x}^{k1} = \tilde{x}_i^{k1} - \tilde{\bar{x}}^{k1}\ v_i^{k1} - \bar{v}^{k1},$$

where $\tilde{\bar{x}}^{k1}$ is the average of the deterministic updates and $\bar{v}^{k1}$ is the average of the velocities. Under the conditions on the velocity parameters, the average velocity tends to zero, and the influence of the velocity difference can be controlled in terms of its norms.

The dual updates contribute to enforcing consensus through the penalty term. The update

$$\lambda_i^{k1} = \lambda_i^k\ \rho\big(41x_i^{k1} - z^{k1}\big)$$

implies that the dual variables accumulate the discrepancies between local positions and the consensus proxy. If the consensus proxies approximate the average position and the velocities are controlled, then the discrepancy decreases over time. The quadratic term in the Lyapunov function penalizes large dual variables and contributes to the overall descent property [42].

The nonconvexity of the local objectives complicates the analysis of convergence to global minima. Instead, attention is focused on stationary points characterized by first-order conditions. A point $x^\star$ is stationary if the zero vector belongs to the sum of the gradients of the convex and nonconvex components at $x^\star$. In the distributed setting, a stationary point corresponds to a collection of local variables $x_i^\star$ that are equal and satisfy

$$0 \in \sum_{i=1}^{N}\big(\nabla g_i x_i^\star\ \partial h_i x_i^\star\big),$$

where ∂h_i denotes a suitable subdifferential [43].

The integrated algorithm can be analyzed under a framework that treats the ADMM step as a proximal gradient mapping and the PSO velocity as a stochastic perturbation. At a high level, the update can be expressed as

$$\mathbf{x}^{k1} = T\mathbf{x}^k\ \mathbf{v}^{k1},$$

where T is a deterministic operator corresponding to the consensus–ADMM mapping, and $\mathbf{v}^{k1}$ represents the stacked velocities. If T is nonexpansive or contractive in the vicinity of a stationary point and the perturbations $\mathbf{v}^{k1}$ form a sequence whose norms vanish or are summable, then the stochastic approximation theory suggests that the sequence $\mathbf{x}^k$ approaches the set of fixed points of T.

The computational complexity of each iteration is determined by the cost of the local primal update, the velocity update, and the consensus mixing. The primal update typically involves evaluating gradients of g_i and h_i and solving a linear system or applying a proximal mapping. If g_i has a simple structure, such as a least squares term, and h_i decomposes across coordinates, the proximal operator may have a closed form [44]. The velocity update involves vector operations that scale linearly with the dimension d. The consensus mixing requires each agent to send and receive a vector of length d to and from its neighbors and to compute a weighted average.

The storage requirements per agent include the current position x_i^k, the dual variable λ_i^k, the velocity v_i^k, and the personal best p_i^k. Additionally, some record of neighborhood best positions may be maintained. The communication complexity per iteration depends on the average degree of the time-varying graph. If the graph is sparse and the degree remains bounded as N grows, the per-iteration communication load per agent is moderate [45].

The design of parameters such as ρ, the step sizes α^k, and the PSO coefficients ω, c_1, and c_2 affects both convergence speed and robustness. A large penalty parameter enforces consensus more strongly but may lead to ill-conditioned local subproblems. Small step sizes improve stability in nonconvex settings but slow down the progress. Larger PSO coefficients increase exploration but can introduce more noise into the consensus process.

Under reasonable assumptions on the graph sequence, the local objectives, and the parameters, it is possible to show that the integrated consensus–ADMM–PSO scheme produces sequences whose expected disagreement converges to zero and whose expected gradients converge to zero in the limit. The precise rates depend on the interplay between the deterministic contraction properties of the ADMM operator and the decay properties of the PSO perturbations [46].

5. Implementation Aspects and Algorithmic Variants

The practical deployment of the consensus–ADMM–PSO scheme on a network of agents involves several design choices related to the representation of local models, the scheduling of communication, the handling of time-varying graphs, and the management of randomization in the PSO component. These choices influence both the computational efficiency and the accuracy of the resulting solutions.

Each agent maintains the local state variables x_i^k, λ_i^k, v_i^k, and p_i^k in its memory. The dimension d of the decision vector influences not only the storage space but also the cost of local linear algebra operations. For high-dimensional variables, it can be beneficial to exploit structure such as sparsity or block separability [47]. For instance, if g_i is a separable sum across coordinates or blocks, the proximal updates may decompose into smaller subproblems that can be processed more efficiently.

The communication between agents is governed by the time-varying graph. In practice, this graph may be induced by physical connectivity, wireless communication ranges, or scheduling policies that activate subsets of links at different times. At each iteration k, agent i must identify its current neighborhood

$$\mathcal{N}_i^k = \{j \mid i, j48 \in \mathcal{E}^k\} \cup \{i\},$$

and then exchange the necessary variables with agents in that set. The usual choice is to exchange the current positions x_j^k and possibly local evaluations of the objective. The consensus proxy

$$y_i^k = \sum_{j \in \mathcal{N}_i^k} w_{ij}^k x_j^k$$

is then computed using the current weight matrix. This operation is linear in the size of the neighborhood and can be implemented using simple summation.

The computation of the neighborhood best position g_i^k for the PSO component can reuse the same communication patterns. Each agent evaluates its local objective at the current position and compares it with the stored values associated with the neighborhood best [49]. When an agent receives updated positions or objective values from its neighbors, it updates g_i^k accordingly. The neighborhood best may be defined over the current neighborhood $\mathcal{N}_i^k$ or over an expanded region that aggregates information over multiple iterations.

Randomization in the PSO velocities is achieved through the generation of random variables r_1^k and r_2^k at each agent. These variables can be sampled independently or derived from pseudo-random generators seeded with different values for each agent. In some implementations, the random coefficients are replaced by deterministic sequences that emulate the exploration properties of PSO while simplifying analysis. The generated velocities may be subjected to clipping, where each component of v_i^k is projected onto a fixed interval to avoid excessively large steps.

The scheduling of local updates can be synchronous or asynchronous. In a synchronous implementation, all agents perform their local primal, velocity, and dual updates at the same iteration index and then exchange variables. This scheme is conceptually simple but requires synchronization across the network [50]. In an asynchronous implementation, agents update their states based on the most recently received information, leading to a more flexible but analytically more complex system. The time-varying graph already introduces variability in which agents are active at different times, and asynchronous updates add an additional layer.

Algorithmic variants can be defined by modifying the order or structure of the updates. One variant places the velocity update before the local augmented Lagrangian step, so that

$$x_i^{k\frac{1}{2}} = x_i^k \; v_i^{k1}$$

is computed first, followed by a deterministic proximal update [51]

$$x_i^{k1} = \mathrm{prox}_{\alpha^k g_i}\big(x_i^{k\frac{1}{2}} - \alpha^k\big(\tilde{\nabla} h_i x_i^{k\frac{1}{2}} \; \lambda_i^k \; \rho x_i^{k\frac{1}{2}} - z^k\big)\big)52.$$

Other variants adjust the relative weighting of the PSO and ADMM components by scaling the velocities or adapting the step sizes. For example, the velocities may be gradually reduced over time, with a decay schedule that decreases the exploration as the algorithm progresses.

The selection of the penalty parameter ρ can be static or dynamic. In a static scheme, ρ is fixed based on prior knowledge of the problem or by preliminary tuning. In a dynamic scheme, ρ may be increased when the consensus violation remains large and decreased when the primal residuals are small [53]. Such adaptations aim to balance the enforcement of consensus with the conditioning of the local subproblems.

Distributed stopping criteria are another aspect of implementation. Since the algorithm is decentralized, no single agent has direct access to global quantities such as the full objective value or the global residuals. Practical stopping rules rely on local information and possibly limited additional communication. For example, each agent can track the change in its local variable norm and the local consensus disagreement and stop when both fall below prescribed thresholds [54]. Alternatively, a fixed number of iterations may be used if strict guarantees are not required.

Fault tolerance and robustness to communication errors can be incorporated by allowing for packet losses, delays, or link failures. In such settings, the weight matrices W^k may occasionally fail to satisfy the ideal stochasticity conditions, and the information used for the consensus proxies may be outdated. The design of the algorithm can accommodate these effects by incorporating resilience mechanisms, such as ignoring stale information beyond a certain age or adjusting the weight matrices based on observed communication quality.

The algorithm can also be specialized to particular classes of local objectives. When the convex component g_i is smooth and strongly convex, simple gradient steps may suffice to approximate the local minimizers [55]. When the nonconvex component h_i has specific structure, such as being a sum of concave penalties applied to linear forms, specialized proximal operators can be used to handle them more accurately. These specializations can improve the convergence speed and the quality of the solutions in particular applications.

6. Numerical Illustration Scenarios and Behavioral Discussion

The behavior of the consensus–ADMM–PSO scheme can be illustrated conceptually by considering representative distributed optimization tasks. While specific numerical values depend on the chosen problem instances and parameter settings, one can describe the qualitative patterns that tend to emerge in typical scenarios.

A first scenario involves distributed learning of a nonlinear model with composite regularization [56]. Each agent i holds a local dataset and seeks to fit a parameter vector x that minimizes a loss function plus a nonconvex regularizer. The convex component $g_i x$ may be a logistic loss, while the nonconvex component $h_i x$ may be a difference of norms that promotes sparsity with reduced bias compared to purely convex penalization. The agents are connected by a time-varying graph modeling intermittent communication. The integrated algorithm is run with a chosen penalty parameter, step sizes, and PSO coefficients [57].

At early iterations, the PSO velocities introduce exploration that allows different agents to explore distinct regions of the parameter space. The consensus penalties and dual variables tend to align the local models, but the velocities maintain diversity among the positions. The local personal bests p_i^k are updated when an agent observes a reduction in its local objective. Over time, the neighborhood bests g_i^k influence the velocities, guiding agents toward regions where other agents have found better solutions. The consensus mixing of positions gradually reduces disagreement across the network.

A second scenario considers a distributed control or resource allocation problem with nonlinear constraints that are handled via penalization. The local objectives capture costs such as deviations from desired trajectories or resource usage, while nonconvex terms represent nonlinear effects or discrete choices that have been relaxed [58]. The communication graph may reflect limited connectivity between controllers or decision makers. The consensus–ADMM–PSO scheme coordinates the local decisions so that they satisfy approximate consensus while exploring feasible and near-feasible regions of the constraint set.

In such a setting, the augmented Lagrangian penalties act as a regularizing force that discourages large disagreements, while the PSO velocities permit agents to temporarily depart from the consensus manifold in search of better trade-offs between cost and constraint satisfaction. The dual variables adjust to reflect persistent disagreements, which in turn modifies the effective local objective landscapes. The interplay between these components can lead to trajectories that oscillate initially and then gradually settle into a configuration that satisfies approximate stationarity [59].

A third scenario involves a nonconvex regression or matrix factorization task distributed across agents that hold different subsets of rows or columns. The local objectives may be nonconvex due to bilinear terms or factorization constraints. The consensus formulation ensures that the global factorization structure is respected, while the PSO component helps avoid poor local minima. The time-varying graph could represent a dynamic topology of computing nodes or a communication network with varying availability.

In this case, the evolution of the collective objective value over iterations often exhibits phases [60]. During initial iterations, the objective may decrease rapidly as the augmented Lagrangian steps exploit the dominant convex structure. The PSO velocities introduce fluctuations that sometimes lead to temporary increases in the objective, but they also enable escapes from shallow local minima. As the velocities are damped or as the algorithm progresses, the trajectory tends to enter a refinement phase where changes in the objective become smaller and the consensus residuals diminish.

The sensitivity of the algorithm to parameter choices can also be discussed. If the inertia weight ω is set close to one and the PSO coefficients c_1 and c_2 are large, the velocities may remain substantial for many iterations, increasing the exploration but also amplifying fluctuations in consensus and objective values [61]. Conversely, if ω is small and the coefficients are modest, the velocities decay quickly, and the algorithm behaves more similarly to a deterministic consensus–ADMM method. The penalty parameter ρ interacts with these settings by controlling the strictness of consensus enforcement.

The time-varying graph structure influences the speed at which information about good solutions propagates through the network. In a graph sequence where links change rapidly but remain jointly connected, neighborhood best information can spread across the network within a moderate number of iterations [62]. In more constrained sequences where connectivity is sparse over long windows, the propagation is slower, and clusters of agents may temporarily hold different neighborhood bests. The consensus mechanism helps to mitigate these disparities, but the transient behavior can be more complex.

An additional aspect concerns the effect of noise and inaccuracies in local objective evaluations. If some agents evaluate their objectives with noise or approximate models, the personal and neighborhood bests may reflect erroneous values. The stochastic nature of the PSO velocities can, in some cases, help to average out such inaccuracies, but persistent bias can affect the stationary points approached by the algorithm [63]. The augmented Lagrangian structure, which is based on local function values and gradients, may be more sensitive to such biases.

Overall, these conceptual numerical scenarios highlight the roles of the different components of the integrated algorithm. The consensus mixing and dual updates aim to reduce disagreements and enforce consistency, the augmented Lagrangian structure reflects the composite nature of the objectives, and the PSO velocities introduce exploratory behavior that can benefit nonconvex optimization. The precise performance in any given application depends on the specifics of the local objectives, the network topology, the time variation of the graph, and the chosen parameters.

7. Conclusion

A distributed optimization framework has been considered in which agents seek to minimize a composite convex–nonconvex objective over a time-varying communication graph [64]. The formulation uses local copies of a global decision vector and

enforces consensus through an augmented Lagrangian approach based on the alternating direction method of multipliers. To address the challenges posed by nonconvex local components and complex landscapes, a particle swarm mechanism is integrated into the primal updates, yielding an algorithm in which each agent maintains local positions, dual variables, and velocities.

The time-varying graph is modeled through a sequence of weight matrices that define consensus mixing steps. Under standard assumptions on joint connectivity and stochasticity, these matrices ensure that consensus can be approached asymptotically. The PSO velocities provide a means of exploring the search space beyond the deterministic trajectories generated by the augmented Lagrangian steps [65]. Their magnitudes and directions are controlled by inertia, cognitive, and social coefficients, as well as by random factors, and they can be tuned to balance exploration and stability.

The convergence discussion focuses on boundedness of iterates, decay of consensus disagreements, and approach to stationary points under reasonable regularity assumptions on the local objectives. The analysis relies on constructing Lyapunov functions that capture the combined influence of local objectives, dual variables, and consensus residuals, and on controlling the stochastic perturbations introduced by the PSO component. While global optimality cannot generally be guaranteed in the presence of nonconvexity, the scheme aims to identify stationary points that satisfy first-order conditions.

Implementation aspects such as communication patterns, parameter selection, storage requirements, and stopping criteria have been outlined [66]. Variants of the basic algorithm can adjust the order of updates, the decay of velocities, and the adaptation of penalty parameters. The conceptual numerical scenarios discussed illustrate the qualitative behavior of the method in representative distributed learning, control, and factorization problems. They highlight how the interplay between consensus, dual ascent, and swarm-based exploration shapes the evolution of the network state.

The development presented here offers a structured way to embed swarm dynamics within an augmented Lagrangian consensus framework for distributed convex–nonconvex programs over time-varying graphs. The resulting algorithm is amenable to further specialization for particular problem classes and to additional analysis aimed at refining convergence guarantees and performance characterizations in specific application domains [67].

Acknowledgments

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

References

[1] X. Feng, X. Liu, and H. Yu, "Group mosquito host-seeking algorithm," *Applied Intelligence*, vol. 44, pp. 665–686, 11 2015.

[2] J. Chen, Y. Xie, H. Chen, Q. Yang, S. Cheng, and Y. Shi, "An improved extremal optimization based on the distribution knowledge of candidate solutions," *Natural Computing*, vol. 16, pp. 135–149, 4 2016.

[3] P. S. Mann and S. Singh, "Energy-efficient hierarchical routing for wireless sensor networks: A swarm intelligence approach," *Wireless Personal Communications*, vol. 92, pp. 785–805, 8 2016.

[4] Q. Kang and H. He, "Task assignment for minimizing application completion time using honeybee mating optimization," *Frontiers of Computer Science*, vol. 7, pp. 404–415, 3 2013.

[5] S. Lu, Y. Wang, H. Liu, M. Miao, and Y. Ma, "Self-assembled ultrathin nanotubes on diamond (100) surface," *Nature communications*, vol. 5, pp. 3666–, 4 2014.

[6] H. Hajipour, H. B. Khormuji, and H. Rostami, "Odma: a novel swarm-evolutionary metaheuristic optimizer inspired by open source development model and communities," *Soft Computing*, vol. 20, pp. 727–747, 11 2014.

[7] 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.

[8] M. Momani and S. Challa, "Survey of trust models in different network domains," *International Journal of Ad hoc, Sensor & Ubiquitous Computing*, vol. 1, pp. 1–19, 9 2010.

[9] M. Goli and S. M. R. Rankoohi, "A new vertical fragmentation algorithm based on ant collective behavior in distributed database systems," *Knowledge and Information Systems*, vol. 30, pp. 435–455, 2 2011.

[10] V. Negru and D. Zaharie, "Introduction to the special issue," *Scalable Computing: Practice and Experience*, vol. 14, 8 2013.

[11] S. Rana, S. Jasola, and R. Kumar, "A hybrid sequential approach for data clustering using k-means and particle swarm optimization algorithm," *International Journal of Engineering, Science and Technology*, vol. 2, 2 2011.

[12] L. M. Correia, A. M. Sebastião, and P. Santana, "On the role of stigmergy in cognition," *Progress in Artificial Intelligence*, vol. 6, pp. 79–86, 12 2016.

[13] M. J. Reddy and D. N. Kumar, "Performance evaluation of elitist-mutated multi-objective particle swarm optimization for integrated water resources management," *Journal of Hydroinformatics*, vol. 11, pp. 79–88, 1 2009.

[14] 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.

[15] Q. Tang, Y. Shen, C. Hu, J. Zeng, and W. Gong, "Swarm intelligence: Based cooperation optimization of multi-modal functions," *Cognitive Computation*, vol. 5, pp. 48–55, 4 2012.

[16] P. Singh, "Favorable trail detection using aco-bellman algorithm in vanets," *International Journal of Modern Education and Computer Science*, vol. 8, pp. 33–39, 1 2016.

[17] B. Salamat and A. M. Tonello, "Stochastic trajectory generation using particle swarm optimization for quadrotor unmanned aerial vehicles (uavs)," *Aerospace*, vol. 4, pp. 27–, 5 2017.

[18] C. Hari and S. S. Tegjyot, "Seepc: A toolbox for software effort estimation using soft computing techniques," *International Journal of Computer Applications*, vol. 31, pp. 12–19, 10 2011.

[19] Y. Wu, F. G. Mármol, and A. Al-Dubai, "Introduction to advances in trust, security, and privacy for wireless networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2013, pp. 287–, 12 2013.

[20] Y. Karpat and T. Özel, "Multi-objective optimization for turning processes using neural network modeling and dynamic-neighborhood particle swarm optimization," *The International Journal of Advanced Manufacturing Technology*, vol. 35, pp. 234–247, 10 2006.

[21] V. R. S. M and R. Ganesan, "Bio-inspired, cluster-based deterministic node deployment in wireless sensor networks," *International Journal of Technology*, vol. 7, pp. 673–, 4 2016.

[22] U. T.S and B. R, "A novel framework for intrusion detection using distributed collaboration detection scheme in packet header data," *International journal of Computer Networks & Communications*, vol. 9, pp. 97–112, 7 2017.

[23] S. A. Kumar, J. Vanualailai, and B. N. Sharma, "Lyapunov-based control for a swarm of planar nonholonomic vehicles," *Mathematics in Computer Science*, vol. 9, pp. 461–475, 10 2015.

[24] 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.

[25] S. Cheng, Q. Qin, J. Chen, and Y. Shi, "Brain storm optimization algorithm: a review," *Artificial Intelligence Review*, vol. 46, pp. 445–458, 2 2016.

[26] M. Duarte, V. Costa, J. Gomes, T. Rodrigues, F. Silva, S. Oliveira, and A. L. Christensen, "Evolution of collective behaviors for a real swarm of aquatic surface robots.," *PloS one*, vol. 11, pp. e0151834–, 3 2016.

[27] A.-D. Ali, M. A. Belal, and M. B. Al-Zoubi, "Load balancing of distributed systems based on multiple ant colonies optimization," *American Journal of Applied Sciences*, vol. 7, pp. 428–433, 3 2010.

[28] M. A. Hsieh, Ádám M. Halász, S. Berman, and V. Kumar, "Biologically inspired redistribution of a swarm of robots among multiple sites," *Swarm Intelligence*, vol. 2, pp. 121–141, 9 2008.

[29] K. Zafar, A. R. Baig, A. Khan, and N. Bukhari, "Optimization of route planning using simulated ant agent system," *International Journal of Computer Applications*, vol. 4, pp. 1–4, 8 2010.

[30] S. Sadeghi-Tabas, A. Akbarpour, M. Pourreza-Bilondi, and S. Samadi, "Toward reliable calibration of aquifer hydrodynamic parameters: characterizing and optimization of arid groundwater system using swarm intelligence optimization algorithm," *Arabian Journal of Geosciences*, vol. 9, pp. 719–, 11 2016.

[31] V. Terziyan, M. Golovianko, and M. Cochez, "International keystone conference - tb-structure: Collective intelligence for exploratory keyword search," *Lecture Notes in Computer Science*, pp. 171–178, 2 2017.

[32] M. Rajeswari, J. Amudhavel, S. Pothula, and P. Dhavachelvan, "Directed bee colony optimization algorithm to solve the nurse rostering problem.," *Computational intelligence and neuroscience*, vol. 2017, pp. 6563498–6563498, 4 2017.

[33] A. Yousif, S. M. Nor, A. H. Abdualla, M. B. Bashir, and S. University-Sudan, "Job scheduling algorithms on grid computing: State-of- the art," *International Journal of Grid and Distributed Computing*, vol. 8, pp. 125–140, 12 2012.

[34] R. Chandrasekar, R. Suresh, and S. Ponnambalam, "Evaluating an obstacle avoidance strategy to ant colony optimization algorithm for classification in event logs," in *2006 International Conference on Advanced Computing and Communications*, pp. 628–629, IEEE, 2006.

[35] C. Guo and B. Zhao, "A pooled-neighbor swarm intelligence approach to optimal reactive power dispatch," *Journal of Zhejiang University-SCIENCE A*, vol. 7, pp. 615–622, 3 2006.

[36] J. Keeney, D. Lewis, and D. O'Sullivan, "Ontological semantics for distributing contextual knowledge in highly distributed autonomic systems," *Journal of Network and Systems Management*, vol. 15, pp. 75–86, 1 2007.

[37] J. Chang and L. Zhang, "Case mix index weighted multi-objective optimization of inpatient bed allocation in general hospital," *Journal of Combinatorial Optimization*, vol. 37, pp. 1–19, 11 2017.

[38] M. LawanyaShri, S. Subha, and B. Balusamy, "Energy-aware fruitfly optimisation algorithm for load balancing in cloud computing environments," *International Journal of Intelligent Engineering and Systems*, vol. 10, pp. 75–85, 2 2017.

[39] V. Trianni and M. Dorigo, "Self-organisation and communication in groups of simulated and physical robots," *Biological cybernetics*, vol. 95, pp. 213–231, 7 2006.

[40] Z. Wei, F. Ge, Y. Lu, L. Li, and Y. Yang, "Chaotic ant swarm for the traveling salesman problem," *Nonlinear Dynamics*, vol. 65, pp. 271–281, 11 2010.

[41] H. Hernandez and C. Blum, "Frogsim: distributed graph coloring in wireless ad hoc networks," *Telecommunication Systems*, vol. 55, pp. 211–223, 8 2013.

[42] J. D. Eversham and V. Ruiz, "Experimental analysis of the reynolds flocking model," *Paladyn, Journal of Behavioral Robotics*, vol. 2, pp. 145–155, 1 2011.

[43] G. Terrazas, C. Cruz, and J. R. González, "Guest editorial: Special issue on nature inspired cooperative strategies for optimization (part ii)," *Memetic Computing*, vol. 3, pp. 229–230, 8 2011.

[44] 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.

[45] X. Cai, Y. Duan, Y. He, J. Yang, and C. Li, "Bee-sensor-c: An energy-efficient and scalable multipath routing protocol for wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 11, pp. 976127–, 3 2015.

[46] C. Gershenson, "Computing networks: A general framework to contrast neural and swarm cognitions," *Paladyn, Journal of Behavioral Robotics*, vol. 1, pp. 147–153, 6 2010.

[47] M. Neshat, A. Adeli, G. Sepidnam, M. Sargolzaei, and A. N. Toosi, "A review of artificial fish swarm optimization methods and applications," *International Journal on Smart Sensing and Intelligent Systems*, vol. 5, pp. 107–148, 1 2012.

[48] S. Wang, Y. Zhang, G. Ji, J. Yang, J. Wu, and L. Wei, "Fruit classification by wavelet-entropy and feedforward neural network trained by fitness-scaled chaotic abc and biogeography-based optimization," *Entropy*, vol. 17, pp. 5711–5728, 8 2015.

[49] K. Dangi and N. Gaud, "A brief review of load balancing issue in cloud computing environment," *International Journal of Computer Applications*, vol. 154, pp. 16–20, 11 2016.

[50] X. Yan, Q. Wu, C. Hu, H. Yao, Y. Fan, Q. Liang, C. Liu, and P. R. China, "Robot path planning based on swarm intelligence," *International Journal of Control and Automation*, vol. 7, pp. 15–32, 7 2014.

[51] Z. Cao, X. Rong, and Z. Du, "An improved brain storm optimization with dynamic clustering strategy," *MATEC Web of Conferences*, vol. 95, pp. 19002–, 2 2017.

[52] S. Orike and D. Corne, "Constrained elitist genetic algorithm for economic load dispatch: Case study on nigerian power system," *International Journal of Computer Applications*, vol. 76, pp. 27–33, 8 2013.

[53] G. G. Riva and J. M. Finochietto, "Icc - pheromone-based in-network processing for wireless sensor network monitoring systems," *Network Protocols and Algorithms*, vol. 4, pp. 6560–6564, 12 2012.

[54] 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.

[55] S. Cheng, B. Liu, T. O. Ting, Q. Qin, Y. Shi, and K. Huang, "Survey on data science with population-based algorithms," *Big Data Analytics*, vol. 1, pp. 1–20, 7 2016.

[56] A. Forestiero, C. Mastroianni, and G. Spezzano, "Building a peer-to-peer information system in grids via self-organizing agents," *Journal of Grid Computing*, vol. 6, pp. 125–140, 2 2007.

[57] V. Pandiri and A. Singh, "Swarm intelligence approaches for multidepot salesmen problems with load balancing," *Applied Intelligence*, vol. 44, pp. 849–861, 11 2015.

[58] A. Goyal and B. Bharti, "A study of load balancing in cloud computing using soft computing techniques," *International Journal of Computer Applications*, vol. 92, pp. 33–39, 4 2014.

[59] H. Zhang, "Distributed intrusion detection model in wireless sensor network," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 11, pp. 61–66, 10 2015.

[60] S. Kaur, R. S. Sawhney, and R. Vohra, "Manet link performance parameters using ant colony optimization approach," *International Journal of Computer Applications*, vol. 47, pp. 40–45, 6 2012.

[61] M. Gupta and D. Gupta, "A new modified firefly algorithm," *International Journal of Recent Contributions from Engineering, Science & IT (iJES)*, vol. 4, pp. 18–23, 7 2016.

[62] Y. Yan and L. A. Osadciw, "Density estimation using a new dimension adaptive particle swarm optimization algorithm," *Swarm Intelligence*, vol. 3, pp. 275–301, 9 2009.

[63] M. Dorigo, M. Birattari, G. A. Di, R. Doursat, A. P. Engelbrecht, L. M. Gambardella, R. Groß, E. Sahin, and T. Stützle, "Ants 2010 special issue," *Swarm Intelligence*, vol. 5, pp. 143–147, 11 2011.

[64] 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.

[65] F. D. Rango, N. Palmieri, X.-S. Yang, and S. Marano, "Swarm robotics in wireless distributed protocol design for coordinating robots involved in cooperative tasks," *Soft Computing*, vol. 22, pp. 4251–4266, 9 2017.

[66] A. Aljanaby, K. R. Ku-Mahamud, and N. M. Norwawi, "Interacted multiple ant colonies optimization approach to enhance the performance of ant colony optimization algorithms," *Computer and Information Science*, vol. 3, pp. 29–34, 1 2010.

[67] D. R. Cañas, A. L. S. Orozco, L. J. G. Villalba, and P.-S. Hong, "Hybrid aco routing protocol for mobile ad hoc networks," *International Journal of Distributed Sensor Networks*, vol. 9, pp. 265485–, 5 2013.