Original Research

Cooperative Ant Colony Optimization for Multi-Commodity Flow and Congestion Management in Software-Defined Networks

Rajendra Maharjan[1] and Kiran Shrestha[2]

[1]Mid-West University, Faculty of Science and Technology, Department of Computer Science, Birendranagar, Surkhet, Nepal.
[2]Far Western University, Department of Computer Science and Information Technology, Mahendranagar, Kanchanpur, Nepal.

Abstract
Software-defined networking decouples the control plane from the data plane and exposes a global view of the network to logically centralized controllers. This programmability enables fine-grained traffic engineering and flow-level optimization that are difficult to realize in traditional distributed routing architectures. At the same time, the growth of heterogeneous traffic and multi-tenant services in data center and wide-area deployments introduces multi-commodity flow patterns with stringent performance and isolation requirements. Congestion management in such environments requires models and algorithms that can exploit centralized visibility while remaining scalable with respect to the number of flows, links, and control epochs. Exact mathematical programming approaches can express these requirements but often become computationally expensive for large networks and short reconfiguration intervals. Metaheuristic approaches can provide approximate solutions within practical time budgets, yet they must be carefully adapted to the structural properties of multi-commodity flow constraints and controller architectures. This paper investigates a cooperative ant colony optimization framework for multi-commodity flow routing and congestion management in software-defined networks. The study combines a linear programming formulation of the traffic engineering objective with a multi-colony ant-based search process that is guided by link-level and path-level pheromone information. Emphasis is placed on the interaction between the linear model and the heuristic components, on strategies for cooperative information sharing among colonies, and on the analysis of congestion-aware pheromone updates under controller resource constraints.

1. Introduction

Software-defined networking introduces a separation between the control and data planes by delegating forwarding decisions to a logically centralized controller that maintains a global representation of the network topology and traffic demands [1]. This architectural shift allows the controller to compute and install forwarding rules that jointly optimize a variety of performance metrics such as latency, throughput, and utilization. In modern deployments, traffic consists of multiple classes and services that can be modeled as commodities sharing the same physical infrastructure. The resulting multi-commodity flow structure raises nontrivial challenges for congestion management because flows belonging to different services interact through shared capacity constraints on links and switches while being subject to heterogeneous performance requirements.

The increasing demand for high-speed data transmission in contemporary networks necessitates advanced strategies for traffic engineering. Software-defined networks facilitate this by offering programmable interfaces that allow for fine-grained control over packet forwarding. In multi-commodity flow scenarios, where diverse applications generate heterogeneous traffic, ensuring equitable resource allocation is paramount. Congestion arises when aggregate flows exceed link capacities, leading to packet drops and increased delays [2]. Mitigation strategies often involve rerouting or load balancing, but these

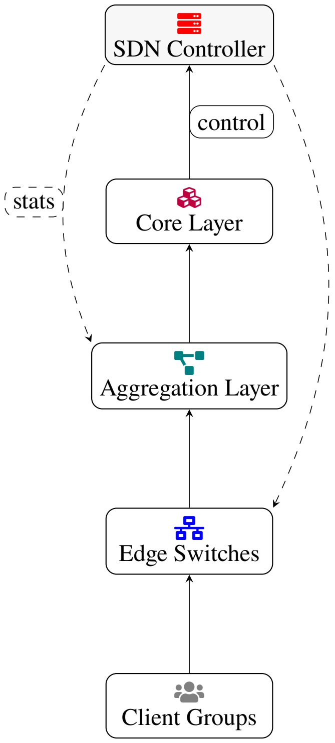

Figure 1: Layered software-defined network architecture showing client groups, forwarding layers, and centralized control.

must be computed efficiently to avoid disrupting ongoing communications. Ant colony optimization mimics natural processes where ants find shortest paths by depositing and following pheromones, translating to probabilistic path selection in artificial systems. In cooperative settings, multiple colonies operate in parallel, with mechanisms for sharing knowledge to avoid local optima traps. This synergy can lead to more robust solutions, particularly in dynamic environments like software-defined networks where topology changes or demand fluctuations occur frequently.

The proposed method builds upon standard ant colony principles by introducing a cooperation layer that synchronizes pheromone matrices at defined intervals. Each colony maintains its own search space but benefits from imported solutions that diversify exploration. Mathematical formulations underpin the algorithm, defining objective functions that penalize high utilization links [3]. Constraints ensure flow conservation and capacity adherence. Simulation platforms such as Mininet are utilized to emulate realistic network behaviors, allowing for controlled experiments. Performance comparisons with baseline algorithms highlight the advantages in terms of reduced congestion metrics. Scalability tests on larger topologies demonstrate the method's viability for enterprise-level deployments. Practical considerations include the overhead of controller-ant interactions and the impact on network stability during optimization phases. The introduction of this cooperative framework seeks to address gaps in current literature by focusing on multi-commodity aspects within software-defined paradigms.

Network operators face ongoing challenges in managing congestion amid growing internet traffic volumes. Software-defined networks provide tools for centralized oversight, enabling proactive adjustments to routing policies [4]. Multi-commodity flows complicate this by introducing interdependencies among different traffic classes, where optimizing one may adversely affect others. Effective management requires algorithms that can handle NP-hard problems approximating optimal distributions. Bio-inspired techniques like ant colony optimization offer promising avenues due to their adaptability and parallelism. Cooperative variants amplify these benefits by fostering information exchange, akin to social insect

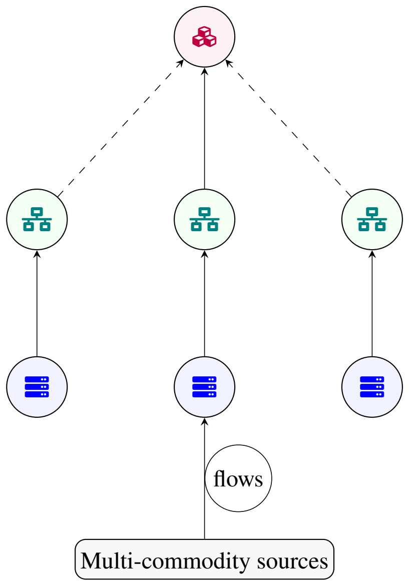

Figure 2: Abstract network substrate used for multi-commodity flows, relating access, aggregation, and core vertices to source commodities.

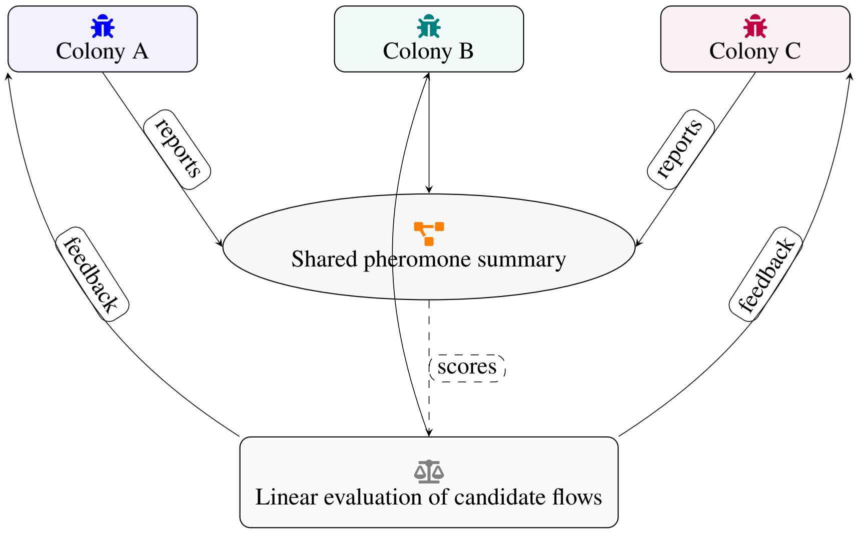

Figure 3: Cooperative ant colonies exchanging pheromone information and receiving linear evaluation feedback for routing decisions.

behaviors where colonies collaborate. This paper details a framework where ants from different colonies deposit pheromones that influence cross-colony decisions, promoting convergence to better global solutions. Key components include initialization of multiple pheromone trails, iterative path construction, and cooperative updates. The integration with software-defined network APIs allows for real-time flow table modifications based on optimized paths [5]. Experimental setups involve synthetic traffic patterns mimicking real-world distributions, with metrics capturing average and peak utilizations. Results from these experiments provide evidence of the framework's efficacy in reducing congestion hotspots. Parameter explorations reveal optimal configurations for colony sizes and cooperation intensities. The work also discusses limitations, such as potential overhead in very large networks, and suggests avenues for

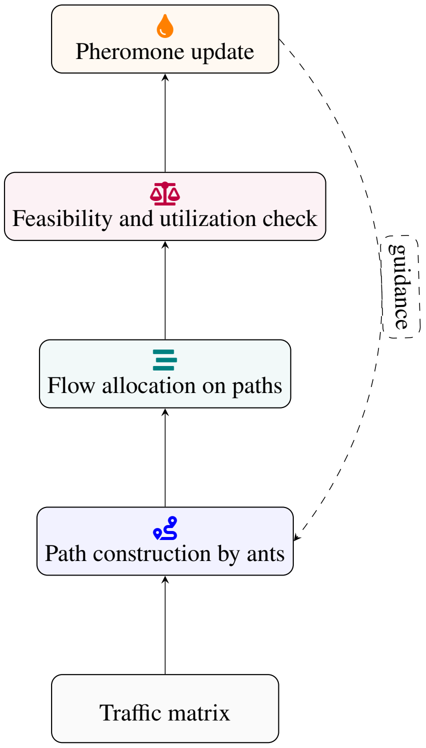

Figure 4: Processing chain from input traffic matrix to pheromone updates, including path construction, allocation, and feasibility checks.

mitigation through hierarchical approaches. By presenting this method, the study contributes to the discourse on intelligent network management.

To expand on the foundational concepts, the cooperative mechanism can be viewed as a form of parallel computing, where each colony processes a subset of the search space independently before merging insights. This parallelization is particularly beneficial in software-defined networks, where controllers can leverage multi-core processors to run colonies concurrently. The pheromone model is extended to include a global component, updated less frequently to maintain diversity [6]. Mathematical modeling of the cooperation includes terms for mixing rates, ensuring stability in updates. The introduction also considers the role of heuristic information, such as inverse delay or residual capacity, in guiding ant decisions. In practice, the framework's ability to handle multi-objective optimization, balancing congestion and delay, is explored through weighted functions. The study's methodology involves rigorous testing, with statistical analysis of results to assess significance. The overall goal is to provide a comprehensive view of how cooperative ant colony optimization can be applied to real-world network problems, offering a balance between theory and practice.

Traditional distributed routing protocols rely on local link-state information and shortest path calculations that typically operate on static or scalar link weights. Such approaches are not well suited for the dynamic and multi-objective optimization problems arising in software-defined networks with time-varying demands, multi-path capabilities, and fine-grained flow definitions [7]. Centralized or hierarchical controllers enable the formulation of traffic engineering problems as global optimization tasks, including linear or mixed-integer programs. However, solving large linear models at high frequency can become demanding for the controller, especially when the number of flows scales with the number of end hosts and services. Approximate algorithms and metaheuristics become attractive in such settings, as they can exploit high-level structure while maintaining practical response times.

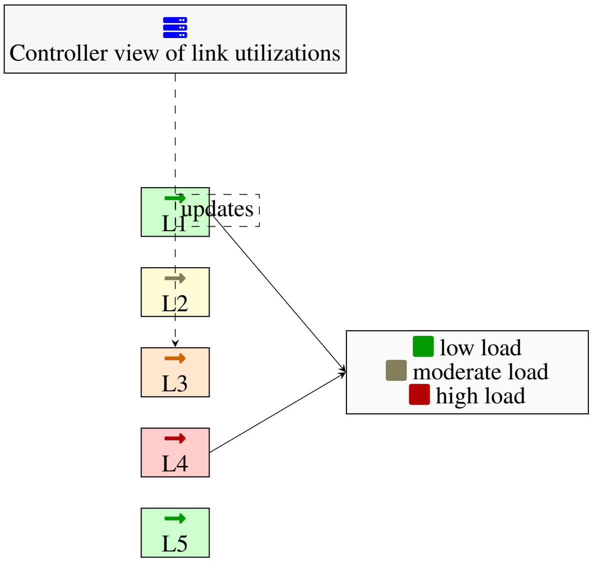

Figure 5: Conceptual link utilization view with qualitative load levels mapped to controller decisions.

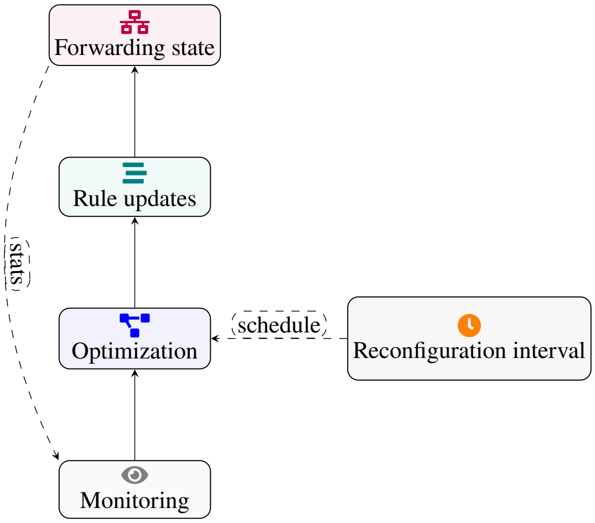

Figure 6: Temporal interaction between monitoring, optimization, and rule deployment across reconfiguration intervals.

Ant colony optimization is a population-based metaheuristic inspired by the collective foraging behavior of ants. Virtual ants traverse the solution space, depositing and reinforcing pheromone traces on promising components. The probabilistic construction of solutions combined with iterative pheromone updates allows the algorithm to approximate good solutions to combinatorial optimization problems such as routing and resource allocation. For multi-commodity flow problems in software-defined networks, ant colony optimization provides a mechanism for exploring alternative path configurations, adapting to changing demands, and incorporating feedback on congestion states.

This work considers a cooperative ant colony optimization framework in which multiple colonies operate in parallel and share information through pheromone structures defined on network links and candidate paths [8]. The framework is coupled with a linear formulation of the multi-commodity flow and congestion management problem that provides a reference model for evaluating and guiding the heuristic search. The study focuses on the design of pheromone update rules that reflect link utilization, on the derivation of linear constraints that capture congestion and controller constraints, and on the evaluation of the algorithm in terms of solution quality and computational characteristics. The analysis is carried out with an emphasis on neutral comparison against baseline approaches such as shortest path routing and simple linear relaxations.

2. Background on Software-Defined Networks and Optimization Techniques

Software-defined networks represent a shift from traditional hardware-centric architectures to software-based control, enabling greater flexibility in traffic management. The core principle involves a centralized controller that communicates with switches via protocols like OpenFlow, dictating forwarding rules dynamically. This setup is particularly advantageous for handling multi-commodity flows, where multiple data streams compete for resources. Congestion management in such environments typically involves monitoring link states and adjusting paths to distribute load evenly [9]. Optimization techniques range from exact methods like integer linear programming to approximations using heuristics. Ant colony optimization, a swarm intelligence algorithm, draws from biological observations of ant foraging, where paths are probabilistically chosen based on pheromone levels and heuristic information. In network contexts, ants simulate packets traversing graphs, updating pheromones to favor low-cost paths. Cooperative ant colony optimization extends this by running multiple independent colonies that periodically share best-found solutions, enhancing diversity and solution quality. This cooperation can take forms such as elite path exchanges or pheromone matrix fusions. In software-defined networks, these algorithms can be implemented at the controller level, leveraging global visibility to compute and enforce routes. The application of ant colony optimization to network problems has evolved over time, with initial focus on single-commodity routing expanding to multi-commodity scenarios. In multi-commodity flows, the challenge lies in satisfying multiple demands without violating capacities, often formulated as minimization of maximum link loads [10]. Cooperative mechanisms address the limitations of single-colony approaches, which may converge prematurely to suboptimal solutions. By allowing colonies to operate on slightly different problem representations or parameter sets, cooperation introduces variability that aids in escaping local minima. Implementation in software-defined networks involves mapping network topologies to graphs, where nodes are switches and edges are links with attributes like capacity and delay. Ants construct solutions by selecting paths for each commodity, evaluating fitness based on congestion metrics. Pheromone updates reinforce successful configurations, while evaporation prevents stagnation. Cooperative intervals are scheduled to balance autonomy and synergy, with empirical tuning determining optimal frequencies. Understanding the mathematical underpinnings is crucial for appreciating these techniques [11]. Network models are represented as directed graphs with capacity constraints on edges. Multi-commodity flow problems seek to route demands while minimizing objectives like total cost or maximum utilization. Ant colony algorithms approximate this through iterative stochastic searches, where transition probabilities guide path choices. Cooperative variants modify update rules to incorporate external inputs, potentially leading to faster convergence. In software-defined networks, real-time data from switches informs these processes, allowing adaptive responses to traffic shifts. Challenges include scaling to large graphs and handling dynamic changes, which cooperative strategies mitigate by distributing computational load. The integration of optimization with network control planes requires careful design to minimize disruptions. Controllers must process optimization outputs efficiently, translating paths into flow rules [12]. Cooperative ant colony methods offer advantages in robustness, as multiple perspectives reduce sensitivity to initial conditions. Performance in congestion management is evaluated through simulations that replicate bursty traffic patterns, measuring metrics like packet loss rates and throughput. These evaluations provide insights into practical viability, highlighting scenarios where cooperation yields measurable benefits over solitary optimization. Expanding the background, software-defined networks have been adopted in data centers for their ability to handle virtualized environments. Optimization techniques in this context often include machine learning for prediction, but ant colony provides a model-free alternative. The cooperative element can be seen as a form of ensemble learning, where diverse solvers combine for better accuracy. Related techniques include bee colony or firefly algorithms, but ant colony's pheromone model suits path-finding [13]. The section reviews historical developments, from Dorigo's original ACO to network applications in the 2000s. Limitations of basic ACO, like slow convergence, are addressed by cooperation. In software-defined networks, security aspects, such as protecting controller communications, are considered in deployments.

3. System Model and Problem Statement

Consider a directed network represented by a graph $G = V, E$ where V is the set of nodes and E is the set of directed links. Each link $i, j \in E$ [14] has an associated capacity $u_{ij} > 0$ and a propagation or transmission cost $c_{ij} \geq 0$. Software-defined switches in the data plane forward packets according to rules installed by a logically centralized controller that has knowledge of G, the link capacities, and current traffic demands.

Traffic is modeled as a set of commodities K [15]. Each commodity $k \in K$ is defined by a source node $s_k \in V$, a destination node $t_k \in V$, and a demand $d_k \geq 0$, [16] which represents the desired rate of flow from s_k to t_k. The controller can split the demand of a commodity across multiple paths. Multi-path routing is assumed to be feasible at the flow granularity supported by the switches and the control protocol, with flow rules allowing forwarding decisions that depend on header fields or flow identifiers.

To model link-level utilization and congestion, associate with each link i, j [17] a utilization variable ρ_{ij} representing the fraction of capacity in use. A congestion management objective often aims to limit the maximum utilization of any link, thereby reducing the risk of queuing delay and packet loss. Introduce an auxiliary variable θ that upper bounds link utilizations. The objective is then to minimize θ [18] subject to flow conservation and capacity constraints while satisfying all demands.

The flow decision variables are denoted by f_{ij}^k, where f_{ij}^k represents the rate of commodity k sent over link i, j. The total flow on link i, j is given by the sum of all commodities using the link. This induces the relation [19]

$${}_{k \in K} f_{ij}^k \leq u_{ij}$$

for all $i, j \in E$. The utilization is $\rho_{ij} = {}_k f_{ij}^k u_{ij}$ and the congestion management objective enforces $\rho_{ij} \leq \theta$ for all links.

Flow conservation is expressed for each commodity at each node. For a commodity k, [20] the net flow at an intermediate node v that is not a source or destination must be zero. At the source node, the net outflow equals d_k, and at the destination node, the net inflow equals d_k [21]. These conditions can be captured by

$${}_{j:v,j \in E} f_{vj}^k - {}_{i:i,v \in E} f_{iv}^k = b_v^k$$

where $b_v^k = d_k$ if $v = s_k$, $b_v^k = -d_k$ if $v = t_k$, and $b_v^k = 0$ otherwise. All flows satisfy non-negativity constraints [22] $f_{ij}^k \geq 0$.

The resulting multi-commodity flow and congestion management problem consists of finding non-negative flows f_{ij}^k and a scalar θ that jointly satisfy the flow conservation constraints, the capacity constraints, and the utilization bounds, while minimizing a suitable function of θ and possibly of link costs. The controller may adopt a reconfiguration interval during which demands are assumed constant and a solution to the optimization problem defines flow rules for that interval [23].

From the perspective of the ant colony optimization framework, the system model specifies the search space and the constraints within which ants construct candidate routing configurations. Each candidate solution can be interpreted as a set of path selections and flow distributions for all commodities. The objective is to approximate solutions that respect capacity and conservation constraints and that yield low values of the congestion indicator θ without requiring exact linear programming solutions at every control epoch.

4. Linear Multi-Commodity Flow and Congestion Formulation

A linear formulation of the multi-commodity flow and congestion management problem can be expressed as follows. The primary objective is to minimize the maximum link utilization θ [24] while obeying capacity and flow conservation constraints. One formulation is

$$\min \ \theta$$

subject to, for all $i, j \in E$,

$${}_{k\in K} f_{ij}^{k} \leq u_{ij}\,\theta$$

and for all $k \in K$, [25] for all $v \in V$,

$${}_{j:v,j\in E} f_{vj}^{k} - {}_{i:i,v\in E} f_{iv}^{k} = b_{v}^{k}.$$

Additionally, enforce

$$f_{ij}^{k} \geq 0$$

for all $i, j \in E$, $k \in K$, and

$$\theta \geq 0$$[26].

In this formulation, θ is a scalar capturing the worst-case relative utilization over all links. Minimizing θ corresponds to spreading traffic in such a way that the most heavily loaded link is as lightly loaded as possible relative to capacity. This type of objective is commonly referred to as min-max or load balancing [27]. It produces solutions with balanced link utilizations, which is a desirable property for congestion management.

To incorporate link costs while still focusing on congestion, a bicriteria or weighted linear objective can be considered. For example, one may minimize a weighted sum of the maximum utilization and the total cost of flows:

$$\min \ \alpha\theta \ \ 1-\alpha \ {}_{k\in K}\,{}_{i,j\in E}\, c_{ij}\, f_{ij}^{k}$$

with $0 \leq \alpha \leq 1$. This expression allows the controller to tune the tradeoff between minimizing the maximum link utilization and minimizing the total cost associated with routing traffic over certain links [28]. The weights c_{ij} may encode delay, loss probability, or administrative preferences.

The linear formulation can also include per-commodity performance constraints if necessary. For example, one may upper bound the total path cost for each commodity k by a parameter H_k, [29] representing a budget on latency or hop count:

$${}_{i,j\in E} c_{ij}\, f_{ij}^{k} \leq H_k\, d_k.$$

This inequality restricts the feasible region to flow patterns that do not exceed the cost budget per unit of demand. It remains linear because both f_{ij}^{k} and d_k are variables or parameters appearing linearly.

In practice, the linear model can be solved by a generic solver when the network and commodity sets are of moderate size. However, solving a large instance at every control interval may not be feasible [30]. The number of variables grows on the order of $|E|\,|K|$, and the number of constraints grows on the order of $|V|\,|K|\ \ |E|$. To reduce complexity, one may consider a path-based formulation in which flows are assigned to a subset of candidate paths for each commodity rather than to arbitrary link combinations. In such a formulation, define P_k [31] as a set of candidate paths for commodity k. Let x_p^k represent the flow of commodity k carried on path $p \in P_k$. The demand constraint becomes [32]

$${}_{p\in P_k} x_p^{k} = d_k$$

for each k, and the link capacity constraints are

$$ {}_{k\in K}\ {}_{p\in P_k:i,j\in p}\, x_p^k \leq u_{ij}\,\theta $$

for all [33] $i, j \in E$. This path-based representation often aligns more closely with how an ant colony algorithm enumerates solutions, since ants naturally construct paths.

The path-based and link-based formulations are equivalent under suitable assumptions, but the path-based formulation only considers a restricted set of candidate paths. The ant colony optimization process can be viewed as a mechanism to explore and refine the candidate path sets P_k. The linear constraints then serve as a guide and an evaluation metric for the candidate solutions generated by the ants. The controller can use the linear model to verify feasibility, compute utilization, and approximate the objective value associated with each configuration proposed by the ant colony framework.

5. Cooperative Ant Colony Optimization Framework

Ant colony optimization models the process of solution construction as a stochastic traversal of a graph where each ant selects edges according to probabilistic rules influenced by pheromone values and heuristic information [34]. In the context of multi-commodity flow routing in software-defined networks, ants generate candidate paths from sources to destinations. The cooperative framework envisaged here operates multiple colonies that share the same underlying physical topology but maintain partially distinct pheromone structures and exploration strategies.

Let τ_{ij} denote the pheromone level associated with link i, j. Pheromone values are initialized to a constant baseline and updated over time based on the quality of solutions discovered. Additional pheromone structures can be defined at the path level. For each commodity [35] k and path index p, a path pheromone ϕ_p^k captures the global desirability of using that path for commodity k. In practice, paths can be represented implicitly as sequences of links, but maintaining explicit path pheromones allows the algorithm to aggregate historical information about successful flow allocations [36].

At each iteration, each colony constructs a set of paths for some or all commodities. The probability that an ant in colony r moves from node i to node j [37] when constructing a path for commodity k is defined by a rule such as

$$ P_{ij}^{k,r} = \frac{\left(\tau_{ij}^{r}\right)^{\alpha} \left(\eta_{ij}^{k}\right)^{\beta}}{{}_{m:i,m\in E} \left(\tau_{im}^{r}\right)^{\alpha} \left(\eta_{im}^{k}\right)^{\beta}} $$

where τ_{ij}^r is the colony-specific pheromone on link i, j, η_{ij}^k is heuristic information for commodity k, [38] and α, β are non-negative parameters controlling the influence of pheromone and heuristic components. The heuristic term η_{ij}^k can encode inverse cost, residual capacity, or an estimate of the contribution of the link to path feasibility. For instance, one may define

$$ \eta_{ij}^{k} = \frac{1}{1\ \ \gamma\,\hat{\rho}_{ij}} $$

where $\hat{\rho}_{ij}$ is an estimate of current utilization and [39] $\gamma \geq 0$ weights the influence of link congestion on the heuristic.

After constructing paths for the commodities, each colony obtains a candidate routing configuration. This configuration is mapped to flows x_p^k and evaluated using the linear model. The evaluation process verifies whether capacity constraints are satisfied and computes the resulting maximum utilization θ [40] and possibly additional metrics such as total cost. If a candidate violates capacity constraints, the algorithm may apply repair strategies or penalize the configuration. The quality of a configuration found by colony r can be quantified by an objective value F^r such as [41]

$$F^r = \lambda\,\theta^r \;\; 1 - \lambda\,C^r$$

where θ^r is the maximum utilization of the configuration, C^r is the total link cost, and $\lambda \in 0, 1$ is a weight parameter [42].

Pheromone updates in each colony combine evaporation and reinforcement. Evaporation reduces pheromone levels to avoid early convergence and to enable exploration of new solutions. For colony r, an evaporation step is

$$\tau_{ij}^r \leftarrow 1 - \rho\,\tau_{ij}^r$$

where $\rho \in 0, 1$ is the evaporation rate [43]. Reinforcement increases pheromone on components used by high-quality configurations. A reinforcement rule can be written as

$$\tau_{ij}^r \leftarrow \tau_{ij}^r \;\; \Delta\tau_{ij}^r$$

where $\Delta\tau_{ij}^r$ is computed from the performance of the current or best configuration of colony r. A simple choice is

$$\Delta\tau_{ij}^r = {}_{k\in K} \frac{\delta_{ij}^{k,r}}{1 \;\; F^r}$$

where $\delta_{ij}^{k,r} = 1$ if link [44] i, j is used by the selected path of commodity k in the configuration of colony r, and $\delta_{ij}^{k,r} = 0$ otherwise.

Cooperation among colonies is introduced through mechanisms that allow some sharing of pheromone information while preserving diversity [45]. One approach is to maintain a global pheromone matrix $\bar{\tau}_{ij}$ that aggregates selected contributions from all colonies. After each iteration, the global pheromone can be updated by

$$\bar{\tau}_{ij} \leftarrow 1 - \xi\,\bar{\tau}_{ij} \;\; \xi\,\frac{{}_r\, w_r\,\tau_{ij}^r}{{}_r\, w_r}$$

where $\xi \in 0, 1$ controls the influence of colonies on the global structure and w_r are weights reflecting the performance of each colony [46]. Colonies may then blend their local pheromone with the global matrix by

$$\tau_{ij}^r \leftarrow \mu\,\tau_{ij}^r \;\; 1 - \mu\,\bar{\tau}_{ij}$$

with a mixing parameter $\mu \in 0, 1$. This cooperative mechanism allows information about good solutions to propagate while preserving differences in exploration strategies or parameter settings among colonies.

Because routing decisions in an SDN must be translated into flow rules, the outputs of the cooperative ant colony framework must be structured in terms of paths and corresponding flow fractions or rates. After each iteration or after a selected number of iterations, the controller can extract the best candidate configuration, verify its feasibility using the linear model, and then generate forwarding rules that implement the flow splitting implied by x_p^k [47]. The frequency of reconfiguration and the number of iterations per epoch are determined by controller capabilities and by the dynamics of traffic demands.

6. Complexity and Convergence Discussion

The complexity of the cooperative ant colony optimization framework depends on several structural parameters of the network and of the algorithm. Key factors include the number of nodes $|V|$, the number of links $|E|$, the number of commodities $|K|$, [48] the number of candidate paths per commodity, the number of colonies, and the number of ants per colony and per iteration. A coarse-grained analysis can be obtained by considering the operations required for solution construction, evaluation, and pheromone updates.

For solution construction, each ant building a path for commodity k performs at most $|V|$ moves in the absence of cycles, because a reasonable implementation prevents revisiting nodes [49]. At each step, the ant evaluates transition probabilities over the outgoing neighbors of the current node. In a sparse network, the average degree is $O1$ relative to $|V|$, so the complexity per path is approximately linear in $|V|$. If each colony constructs paths for all commodities and uses [50] A ants per commodity, the complexity per iteration for solution construction scales as

$$O\big(A\,|K|\,|V|\big)\,.$$

Evaluation of each candidate configuration involves calculating link loads, utilizations, and objective values. Given the path flows [51] x_p^k, the total flow on each link i, j is obtained by summing over all commodities and paths using that link. In the worst case, if each path includes a substantial fraction of the links, the evaluation cost per configuration is

$$O\big(|E|\,|K|\big)$$ 52

although in practice it can be reduced by exploiting sparsity and precomputed path-link incidence relations. Because metaheuristic algorithms typically evaluate multiple configurations per iteration, the overall evaluation cost can be a significant component of runtime. However, compared to solving a full linear program, the evaluation of candidate configurations is relatively lightweight.

Pheromone updates require iterating over the links used by selected configurations and adjusting pheromone values according to the reinforcement rules. Assuming that only a subset of best configurations per colony contributes to pheromone reinforcement, the complexity of updates is bounded by the number of links appearing in those configurations. This cost is also linear in the size of the network and the number of commodities, particularly if the number of reinforcing configurations is small.

The convergence properties of ant colony optimization algorithms are typically studied in terms of stochastic processes and Markov chain theory [53]. Under certain conditions on evaporation and reinforcement parameters, the pheromone updates define a stochastic process that has absorbing states corresponding to deterministic solutions. For example, if evaporation is strictly positive and reinforcement is applied only to globally best solutions, the algorithm can, in theory, converge to a fixed routing configuration. A simplified analysis considers the probability that an ant constructs a solution with objective value at most z as a function of iteration number. Assuming the presence of at least one configuration with objective value at most $z^\star$, and that reinforcement increases the probability of reconstructing such configurations, one can express a recurrence relation for the expected proportion of ants constructing near-optimal solutions [54].

A qualitative picture emerges from considering that evaporation gradually reduces pheromone levels on links not used in good configurations, while reinforcement increases pheromone on links frequently used by good configurations. Over time, the joint effect biases the transition probabilities in favor of links belonging to frequently reinforced paths, which increases the probability that future ants reconstruct similar solutions. However, if evaporation is too slow or reinforcement too strong, the algorithm may converge prematurely to suboptimal configurations. Selecting parameters ρ, ξ, μ, [55] and weighting factors carefully is therefore important.

The cooperative multi-colony structure can influence convergence in both positive and negative ways. On one hand, multiple colonies exploring different regions of the solution space can delay premature convergence by maintaining diverse pheromone landscapes. On the other hand, if the global pheromone aggregation mechanism is too aggressive, colonies may rapidly synchronize and lose diversity, reducing the benefits of parallel exploration. The mixing parameter μ and the aggregation parameter ξ [56] control this tradeoff. For moderate values of these parameters, colonies exchange information about good solutions while preserving sufficient independence to explore alternative configurations.

In the context of software-defined networks, the convergence behavior of the algorithm is also constrained by operational requirements. The controller may allocate only a limited number of iterations before it must commit to a routing configuration for the next reconfiguration interval. Therefore, convergence in a strict theoretical sense may not be reached in practice. Instead, the algorithm aims to improve candidate solutions steadily over iterations and to provide configurations with acceptable performance within the allotted time. Empirical evaluation can characterize the evolution of maximum link utilization and total cost as a function of iteration count and help determine suitable parameter settings and iteration budgets [57].

7. Numerical Experiments in Software-Defined Networks

Numerical experiments can be used to examine the behavior of the cooperative ant colony optimization framework on software-defined network topologies with multi-commodity traffic. Although real deployments may involve very large and dynamic networks, controlled experiments on representative topologies provide insight into the capacity of the algorithm to balance congestion and path cost, as well as its sensitivity to parameter settings and demand patterns.

A typical experimental setup considers a network emulator or simulator with a fixed topology during a set of runs. The topology can be a fat-tree, a spine-leaf configuration, or a general mesh representing a wide-area backbone. Each link is assigned a capacity u_{ij} and a cost c_{ij}. Traffic demands are generated between randomly selected node pairs or according to specific traffic matrices that emulate data center or wide-area patterns. Various load levels can be studied by scaling all demands by a factor [58] δ so that total offered load increases from low utilization regimes to high utilization regimes.

For each demand matrix and load level, the cooperative ant colony framework is executed for a prescribed number of iterations and with given values of algorithmic parameters such as the number of colonies, number of ants, evaporation rate, and heuristic weighting. At each iteration, the best candidate configuration is recorded along with the corresponding maximum utilization θ and total cost. For comparison, baseline configurations can be obtained by computing shortest path routing with respect to link cost, by solving the linear congestion minimization problem exactly when feasible, or by applying simpler heuristics that distribute flows proportionally to link capacities [59].

The resulting performance indicators allow the construction of metrics such as the relative gap in maximum utilization between the heuristic solution and the reference linear programming solution, or the average reduction in maximum utilization compared to shortest path routing. For example, one can compute a relative gap

$$G_\theta = \frac{\theta^{\mathrm{ACO}} - \theta^{\mathrm{LP}}}{\theta^{\mathrm{LP}}}$$

where θ^{ACO} is the minimum maximum utilization achieved by the cooperative ant colony framework within the allowed iterations and θ^{LP} is the optimal maximum utilization obtained from the linear program when solvable. A similar measure can be defined for total cost. When the linear program cannot be solved due to size or time limitations, the comparison may be restricted to heuristic baselines.

Another aspect of interest is the variability of performance across different random seeds, which reflects the stochastic nature of the algorithm [60]. By executing multiple runs with different initial pheromone states or random number streams, one can estimate empirical distributions of θ^{ACO} and C^{ACO}. The spread of these distributions indicates the robustness of the algorithm to stochastic fluctuations. It is often observed that cooperation among colonies reduces variance by enabling information sharing, although the extent of reduction depends on parameter settings.

The experiments can also measure computational indicators such as runtime per iteration, total runtime per configuration, and the fraction of controller resources consumed by the optimization process. Because software-defined controllers must manage rule installation, monitoring, and other control tasks in addition to optimization, it is important that the metaheuristic does not saturate computational capacity. Empirical measurements allow the identification of feasible combinations of colony count, ant count, and iteration count that respect processing constraints while delivering acceptable solutions.

In addition to aggregate metrics, experiments can examine spatial patterns of congestion and routing [61]. For example, link utilization distributions can be computed under different routing strategies. The cooperative ant colony framework may reduce the frequency of links operating above a specified utilization threshold such as 80% of capacity, by reassigning flows to alternative paths even if such paths are slightly longer in terms of cost. Visualizations of utilization distributions and path allocations can help to understand how the algorithm trades off utilization and cost.

Finally, experiments can consider dynamic scenarios in which demand matrices change over time and the algorithm is re-invoked at each reconfiguration interval. In such scenarios, one can measure how quickly the cooperative ant colony framework adapts to changes in traffic patterns and whether successive configurations differ significantly, which may impact the stability of flow rules and the overhead of updating switches. Smoothness constraints or penalties for changing routing decisions can be integrated into the evaluation function to balance adaptation and stability [62]. The flexibility of the metaheuristic allows such extensions, and experimental results can illustrate the tradeoffs between responsiveness to demand changes and the desire to minimize reconfiguration overhead.

8. Conclusion

This paper has examined a cooperative ant colony optimization framework for multi-commodity flow routing and congestion management in software-defined networks. A linear formulation of the multi-commodity flow problem with a min-max utilization objective has been used to express flow conservation, capacity constraints, and link utilization measures, and to evaluate candidate routing configurations. The cooperative framework organizes multiple colonies that construct end-to-end paths for commodities, exchange information through pheromone structures, and adapt their transition probabilities based on link congestion and path quality indicators. The interplay between link-level and path-level pheromone values, combined with heuristic estimates of residual capacity, guides the search toward configurations that balance link utilization and path cost.

The discussion has highlighted computational aspects of the framework and its relationship to exact linear programming approaches. While linear models provide a clear and flexible reference for expressing constraints and objectives, their direct solution may not always be practical at the scales and frequencies required by software-defined controllers operating under real-time constraints [63]. The ant colony approach offers a complementary perspective in which approximate solutions are generated iteratively, with complexity determined largely by the number of colonies, ants, and iterations that the controller is prepared to allocate. Cooperative mechanisms among colonies allow the sharing of information about promising routing patterns while preserving diversity in exploration.

From the standpoint of practical deployment, numerical experiments on representative topologies and traffic patterns are important to quantify how the cooperative ant colony framework behaves under different load levels, demand distributions, and parameter settings. Such experiments can examine performance metrics including maximum link utilization, total cost, variability across runs, and computational load on the controller, as well as dynamic properties such as adaptation to changing traffic and stability of routing decisions. The framework can also be extended to incorporate additional constraints or objectives, such as quality-of-service requirements for specific traffic classes, or limits on the frequency and magnitude of reconfigurations.

Overall, the combination of a linear multi-commodity flow model with a cooperative ant colony optimization process provides a structured approach to exploring the space of feasible routing configurations in software-defined networks. The linear model serves both as a specification of constraints and as a tool for evaluating candidate solutions, while the metaheuristic offers a flexible means of obtaining feasible configurations within given time budgets. Further work can refine the interaction between the model and the heuristic, explore alternative forms of cooperation among colonies, and investigate the behavior of the framework on larger and more heterogeneous networks with varying controller architectures and monitoring capabilities [64].

References

[1] G. ning Xu, S. yun Huang, Q. song Qi, and Z. yuan Kang, “A new bionic swarm intelligence optimization: Construction and application of modified moth-flame optimization algorithm,” *DEStech Transactions on Engineering and Technology Research*, 5 2017.

[2] M. M. al Rifaie, “Penguins huddling optimisation,” *International Journal of Agent Technologies and Systems*, vol. 6, pp. 1–29, 4 2014.

[3] T. V. V. Kumar, A. Kumar, and R. Singh, “Distributed query plan generation using particle swarm optimization,” *International Journal of Swarm Intelligence Research*, vol. 4, pp. 58–82, 7 2013.

[4] R. Chandrasekar and S. Misra, “Introducing an aco based paradigm for detecting wildfires using wireless sensor networks,” in *2006 International Symposium on Ad Hoc and Ubiquitous Computing*, pp. 112–117, IEEE, 2006.

[5] J. ze Sun, S. yan Wang, and H. Chen, “A guaranteed global convergence social cognitive optimizer,” *Mathematical Problems in Engineering*, vol. 2014, pp. 1–8, 9 2014.

[6] V. Trianni, E. Tuci, K. M. Passino, and J. A. R. Marshall, “Swarm cognition: an interdisciplinary approach to the study of self-organising biological collectives,” *Swarm Intelligence*, vol. 5, pp. 3–18, 12 2010.

[7] J. M. M. Vazquez, J. A. Ramírez, L. Gonzalez-Abril, and F. V. Morente, “Designing adaptive learning itineraries using features modelling and swarm intelligence,” *Neural Computing and Applications*, vol. 20, pp. 623–639, 2 2011.

[8] C. Kolias, V. Kolias, and G. Kambourakis, “Termid: a distributed swarm intelligence-based approach for wireless intrusion detection,” *International Journal of Information Security*, vol. 16, pp. 401–416, 6 2016.

[9] A. Hadjimichael, J. Comas, and L. Corominas, “Do machine learning methods used in data mining enhance the potential of decision support systems? a review for the urban water sector,” *AI Communications*, vol. 29, pp. 747–756, 12 2016.

[10] S. Salcedo-Sanz, L. Carro-Calvo, M. M. Claramunt, A. Castañer, and M. Mármol, “Effectively tackling reinsurance problems by using evolutionary and swarm intelligence algorithms,” *Risks*, vol. 2, pp. 132–145, 4 2014.

[11] S. Nebti and A. Boukerram, “Handwritten characters recognition based on nature-inspired computing and neuro-evolution,” *Applied Intelligence*, vol. 38, pp. 146–159, 6 2012.

[12] S. Menaka and M. Jayanthi, “Intelligent routing using ant algorithms for wireless ad hoc networks,” *International Journal of Computer Network and Information Security*, vol. 5, pp. 51–57, 8 2013.

[13] 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.

[14] M. Saska, V. Vonásek, J. Chudoba, J. Thomas, G. Loianno, and V. Kumar, “Swarm distribution and deployment for cooperative surveillance by micro-aerial vehicles,” *Journal of Intelligent & Robotic Systems*, vol. 84, pp. 469–492, 2 2016.

[15] V. Gokul and S. Elias, “Parallel implementation of cross-layer optimization - a performance evaluation based on swarm intelligence,” *ICTACT Journal on Soft Computing*, vol. 02, pp. 285–290, 1 2012.

[16] A. I. Saleh, “An efficient system-oriented grid scheduler based on a fuzzy matchmaking approach,” *Engineering with Computers*, vol. 29, pp. 185–206, 1 2012.

[17] R. Janapati, C. Balaswamy, and K. Soundararajan, “Enhancement of indoor localization in wsn using pso tuned ekf,” *International Journal of Intelligent Systems and Applications*, vol. 9, pp. 10–17, 2 2017.

[18] S. Nair, “Fuzzy logic based parameter adaptation of interior search algorithm,” *International Journal of Trend in Scientific Research and Development*, vol. Volume-1, pp. 1281–1288, 8 2017.

[19] A. J. Mohammed, Y. Yusof, and H. Husni, “Gf-clust: A nature-inspired algorithm for automatic text clustering,” *Journal of Information and Communication Technology*, vol. 15, pp. 57–81, 5 2016.

[20] T. Stirling, S. Wischmann, and D. Floreano, “Energy-efficient indoor search by swarms of simulated flying robots without global information,” *Swarm Intelligence*, vol. 4, pp. 117–143, 2 2010.

[21] Y. Wu, S. Kiviluoto, K. Zenger, X.-Z. Gao, and X. Huang, “Hybrid swarm algorithms for parameter identification of an actuator model in an electrical machine,” *Advances in Acoustics and Vibration*, vol. 2011, pp. 1–12, 5 2011.

[22] null null, “Correction: Evolution of self-organized task specialization in robot swarms.,” *PLoS computational biology*, vol. 12, pp. e1004996–e1004996, 6 2016.

[23] 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.

[24] W.-A. Yang, Y. Guo, and W. Liao, “Optimization of multi-pass face milling using a fuzzy particle swarm optimization algorithm,” *The International Journal of Advanced Manufacturing Technology*, vol. 54, pp. 45–57, 9 2010.

[25] B. Gu, X. Hong, and P. Wang, “Analysis for bio-inspired thrown-box assisted message dissemination in delay tolerant networks,” *Telecommunication Systems*, vol. 52, pp. 217–227, 8 2011.

[26] R. O’Grady, R. Gro, A. L. Christensen, and M. Dorigo, “Self-assembly strategies in a group of autonomous mobile robots,” *Autonomous Robots*, vol. 28, pp. 439–455, 2 2010.

[27] F. NarinNur, N. N. Moon, and N. R. Chakraborty, “A survey on routing protocols in wireless multimedia sensor networks,” *International Journal of Computer Applications*, vol. 73, pp. 41–46, 7 2013.

[28] S. M. Girirajkumar, K. Ramkumar, and S. O. Sarma, “Real time application of ants colony optimization,” *International Journal of Computer Applications*, vol. 3, pp. 34–46, 6 2010.

[29] D. J. Sathya, “Automatic brain mr image lesion segmentation using artificial bee colony optimization algorithm,” *International Journal of Computer Applications*, vol. 163, pp. 28–33, 4 2017.

[30] Z. Yao and Z. Ren, “Path planning for coalmine rescue robot based on hybrid adaptive artificial fish swarm algorithm,” *International Journal of Control and Automation*, vol. 7, pp. 1–12, 8 2014.

[31] H. Huang and T. Zhuo, “Multi-model cooperative task assignment and path planning of multiple ucav formation,” *Multimedia Tools and Applications*, vol. 78, pp. 415–436, 6 2017.

[32] 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.

[33] K. . and P. Verma, “Clustering amelioration and optimization with swarm intelligence for color image segmentation,” *International Journal of Database Theory and Application*, vol. 8, pp. 51–64, 10 2015.

[34] Z. Yang, M. Emmerich, T. Bäck, and J. N. Kok, “Integrated computer-aided engineering - multi-objective inventory routing with uncertain demand using population-based metaheuristics,” *Integrated Computer-Aided Engineering*, vol. 23, pp. 205–220, 6 2016.

[35] H. F. Eid, “Performance improvement of plant identification model based on pso segmentation,” *International Journal of Intelligent Systems and Applications*, vol. 8, pp. 53–58, 2 2016.

[36] A. Ayari and S. Bouamama, “Rcar - a new multi-robot path planning algorithm: Dynamic distributed particle swarm optimization,” *Robotics and biomimetics*, vol. 4, pp. 8–8, 11 2017.

[37] K. A. Fakeeh, “Developing virtual class room models with bio inspired algorithms for e-learning: A survey for higher technical education for saudi arabia vision 2030,” *International Journal of Applied Information Systems*, vol. 12, pp. 8–21, 11 2017.

[38] T. T. Khuat and M. H. Le, “A genetic algorithm with multi-parent crossover using quaternion representation for numerical function optimization,” *Applied Intelligence*, vol. 46, pp. 810–826, 11 2016.

[39] P. Khurana and I. K. Aulakh, “Wireless sensor network routing protocols: A survey,” *International Journal of Computer Applications*, vol. 75, pp. 17–25, 8 2013.

[40] H. Weihua, M. Zhong, D. Xinfa, G. Yi, and X. Mingdi, “Cluster load balancing algorithm based on intelligent genetic algorithm of vector,” *International Journal of Grid and Distributed Computing*, vol. 10, pp. 73–84, 2 2017.

[41] M. Woźniak, D. Połap, C. Napoli, and E. Tramontana, “Application of bio-inspired methods in distributed gaming systems,” *Information Technology and Control*, vol. 46, pp. 150–164, 3 2017.

[42] 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.

[43] V. S. Borkar and D. Das, “A novel aco algorithm for optimization via reinforcement and initial bias,” *Swarm Intelligence*, vol. 3, pp. 3–34, 12 2008.

[44] F. Yang, J. Li, T. Lei, and S. Wang, “Architecture and key technologies for internet of vehicles:a survey,” *Journal of Communications and Information Networks*, vol. 2, pp. 1–17, 6 2017.

[45] A. John, A. Schadschneider, D. Chowdhury, and K. Nishinari, “Characteristics of ant-inspired traffic flow: Applying the social insect metaphor to traffic models,” *Swarm Intelligence*, vol. 2, pp. 25–41, 4 2008.

[46] L. Mohammadpour, M. Hussain, A. Aryanfar, V. M. Raee, and F. Sattar, “Evaluating performance of intrusion detection system using support vector machines: Review,” *International Journal of Security and Its Applications*, vol. 9, pp. 225–234, 9 2015.

[47] S. N. Devi and A. Pethalakshmi, “Application of aco for resource discovery in grid computing environment,” *International Journal of Computer Applications*, vol. 43, pp. 13–16, 4 2012.

[48] W. Xuewu, L. Xue, Y. Yan, and X. Gu, “Welding robot collision-free path optimization,” *Applied Sciences*, vol. 7, pp. 89–, 2 2017.

[49] N. Bessis and E. Asimakopoulou, “Preface for a special issue on “smart environments and collective computational intelligence for disaster management”,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 4, pp. 533–534, 12 2012.

[50] J. Handl and B. Meyer, “Ant-based and swarm-based clustering,” *Swarm Intelligence*, vol. 1, pp. 95–113, 11 2007.

[51] 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.

[52] D. Tian, H. Junjie, Z. Sheng, Y. Wang, J. Ma, and J. Wang, “Swarm intelligence algorithm inspired by route choice behavior,” *Journal of Bionic Engineering*, vol. 13, pp. 669–678, 12 2016.

[53] L. Ma, X. Wang, R. Yu, G. Yang, J. Li, and M. Huang, “Biomimicry of plant root growth using bioinspired foraging model for data clustering,” *Neural Computing and Applications*, vol. 29, pp. 819–836, 7 2016.

[54] P. Sengottuvelan and N. Prasath, “Bafsa: Breeding artificial fish swarm algorithm for optimal cluster head selection in wireless sensor networks,” *Wireless Personal Communications*, vol. 94, pp. 1979–1991, 4 2016.

[55] L. Li, A. Martinoli, and Y. S. Abu-Mostafa, “Learning and measuring specialization in collaborative swarm systems,” *Adaptive Behavior*, vol. 12, pp. 199–212, 12 2004.

[56] R. Kaur and S. Angurala, “Improving displacement number and overheads of drfn using artificial bee colony technique in wsns,” *International Journal of Computer Applications*, vol. 131, pp. 19–23, 12 2015.

[57] P. K. Sinha and S. R. Dhore, “Multi-agent optimized load balancing using spanning tree for mobile services,” *International Journal of Computer Applications*, vol. 1, pp. 35–42, 2 2010.

[58] V. Teja and R. G. Mishra, “Application of wireless sensor networks in robotics (swarm intelligence),” *Zenodo (CERN European Organization for Nuclear Research)*, 2 2016.

[59] T. Srinivasan and B. Palanisamy, “Scalable clustering of high-dimensional data technique using spcm with ant colony optimization intelligence,” *TheScientificWorldJournal*, vol. 2015, pp. 107650–107650, 10 2015.

[60] N. R. Nikhil and P. Dudi, “Applications of swarm intelligence techniques in grid computing,” *Indian Journal of Science and Technology*, vol. 9, 12 2016.

[61] T. Srinivasan, V. Vijaykumar, and R. Chandrasekar, “An auction based task allocation scheme for power-aware intrusion detection in wireless ad-hoc networks,” in *2006 IFIP International Conference on Wireless and Optical Communications Networks*, pp. 5–pp, IEEE, 2006.

[62] null Meenu, “Taxonomy of nature inspired computational intelligence in digital image processing for harsh weather,” *International Journal of Advanced Research in Computer Science*, vol. 8, pp. 15–21, 10 2017.

[63] X. Shan and H. Cheng, “Modified bat algorithm based on covariance adaptive evolution for global optimization problems,” *Soft Computing*, vol. 22, pp. 5215–5230, 12 2017.

[64] R. Kumar and T. Prashar, “Performance analysis of load balancing algorithms in cloud computing,” *International Journal of Computer Applications*, vol. 120, pp. 19–27, 6 2015.