CSE 4106: Computer Networks Laboratory

AETHER — Adaptive Exploration Through Heuristic Routing

Submitted By:

Choyan Mitra Barua Bijoy
Roll: 2007101

Submitted To:

Md. Shakhawat Hossain
Lecturer

&

Farhan Sadaf
Lecturer

Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna 9203, Bangladesh

October, 2025

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Project Overview

AETHER—Adaptive Exploration Through Heuristic Routing—is a bio-inspired network routing framework that solves the problem of inefficient and static routing in computer networks. Traditional routing protocols rely on fixed routing rules that cannot adapt to changing network conditions. AETHER introduces an intelligent, self-learning approach based on Ant Colony Optimization (ACO) principles to discover and optimize routing paths dynamically. The project implements a complete ACO-based routing system in the OMNeT++ network simulator, where virtual ant agents explore the network, communicate through pheromone signals, and collectively learn the optimal routes for packet forwarding. Unlike static routing tables, AETHER continuously adapts to network topology, link failures, and congestion through the emergent behavior of distributed ant agents.

1.2 Motivation

Traditional network routing algorithms suffer from several drawbacks: they use static routing tables that cannot respond to network congestion or changes, depend on a single optimal path which can become a bottleneck, are unable to distribute load efficiently across available links, and offer limited fault tolerance when a link fails. AETHER addresses these issues by introducing adaptive routing that learns multiple high-quality paths, balances network traffic intelligently, and can discover alternative routes as network conditions change.

1.3 Key Contributions

This project makes the following contributions to network routing research:

1. **ACO-Based Dynamic Routing:** Implements a complete Ant Colony Optimization algorithm for discovering optimal routing paths through network exploration.
2. **Congestion-Aware Load Balancing:** Introduces a congestion penalty mechanism that reduces pheromone on heavily-used links, naturally distributing traffic across the network.
3. **Probabilistic Routing:** Combines deterministic (Greedy) and probabilistic (Roulette Wheel) routing methods to balance exploitation of good paths with exploration of alternatives.
4. **Practical Implementation:** Demonstrates ACO routing in a realistic OMNeT++ environment with complete packet forwarding, loop detection, and fault tolerance.

1.4 Objectives

The main objectives of AETHER are:

- To implement a fully functional ACO-based routing algorithm in OMNeT++ that learns optimal paths through agent exploration.
- To evaluate the performance of two routing methods Greedy Local and Probabilistic and compare their effectiveness.
- To implement congestion-aware routing that balances traffic load across network links.
- To demonstrate fault tolerance through loop detection, backtracking, and alternative path discovery.
- To provide a foundation for future research in bio-inspired network routing and adaptive algorithms.

Chapter 2

Background

2.1 Ant Colony Optimization (ACO)

2.1.1 Biological Inspiration

Ant Colony Optimization is a bio-inspired metaheuristic algorithm based on the foraging behavior of real ants. In nature, ants searching for food initially wander randomly. When an ant discovers food, it returns to the colony while depositing a chemical substance called *pheromone* along its path. Other ants detect this pheromone trail and are more likely to follow it, reinforcing the trail with their own pheromone deposits.

Over time, shorter paths accumulate pheromone faster because ants complete the journey more quickly, leading to more frequent reinforcement. Meanwhile, pheromone on longer paths evaporates before being reinforced. This positive feedback mechanism allows the ant colony to collectively discover the shortest path between the nest and food source without centralized control.

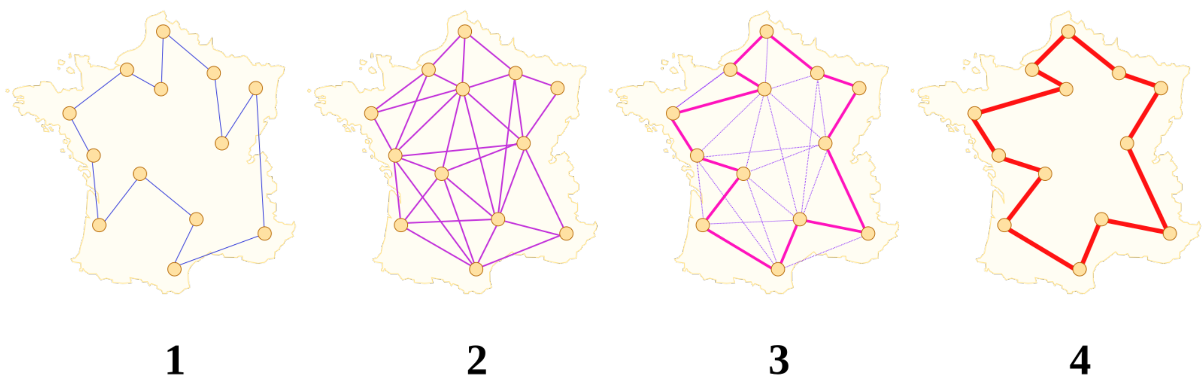

Figure 2.1: 3-Router Triangle Network Topology

2.1.2 ACO for Network Routing

In network routing, ACO translates biological ant behavior into virtual agents that explore network paths:

- **Ants:** Software agents that traverse the network from source to destination.
- **Pheromone:** Virtual chemical signal stored on network links representing path quality.
- **Nest:** Source router where ants originate.
- **Food:** Destination router that ants seek.
- **Path:** Sequence of network links connecting source to destination.

2.1.3 ACO Algorithm Phases

The ACO algorithm for routing consists of three main phases executed iteratively:

Phase 1: Forward Ant Exploration

Forward ants are launched from source routers and explore paths to destination routers. At each intermediate router, the ant probabilistically selects the next hop based on:

1. **Pheromone intensity** τ_{ij}: The accumulated pheromone on link $(i \to j)$.
2. **Heuristic visibility** η_{ij}: The inverse of link cost, calculated as $\eta_{ij} = \frac{1}{d_{ij}}$, where d_{ij} is the transmission delay.

The probability that an ant at node i selects neighbor j as the next hop is:

$$p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}]^\beta} \tag{2.1}$$

where:

- $p_{ij}^k(t)$: Probability that ant k moves from node i to j at time t.
- $\tau_{ij}(t)$: Pheromone level on link $(i \to j)$ at time t.
- η_{ij}: Visibility (heuristic desirability) of link $(i \to j)$.
- α: Pheromone importance weight (controls exploitation).
- β: Visibility importance weight (controls exploration based on cost).
- $\mathcal{N}_i^k$: Set of feasible neighbors for ant k at node i (unvisited nodes).

As the forward ant traverses the network, it records:

- Visited nodes to prevent loops.
- Total path cost (sum of link delays).
- Complete path from source to destination.

Phase 2: Backward Ant Pheromone Deposition

Once a forward ant reaches its destination, it converts into a *backward ant* and retraces the discovered path back to the source. Along this return journey, the backward ant deposits pheromone on each link proportional to the path quality:

$$\Delta\tau_{ij}^k = \frac{Q}{L_k} \tag{2.2}$$

where:

- $\Delta\tau_{ij}^k$: Pheromone deposited by ant k on link $(i \to j)$.
- Q: Pheromone deposit constant (typically 100).
- L_k: Total cost of the path discovered by ant k.

The pheromone update rule for each link is:

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \Delta\tau_{ij}^k \tag{2.3}$$

This mechanism ensures that shorter paths (lower L_k) receive higher pheromone deposits, making them more attractive to future ants.

Phase 3: Pheromone Evaporation

After all ants complete their journeys in an iteration, pheromone evaporation is applied to all links:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) \tag{2.4}$$

where:

- ρ: Evaporation rate ($0 < \rho < 1$), typically 0.1 (10% evaporation).

Evaporation serves two purposes:

- **Prevents stagnation:** Avoids excessive accumulation on a single path.
- **Forgets bad paths:** Allows the algorithm to explore new routes if conditions change.

To ensure pheromone never drops to zero, a minimum threshold is enforced:

$$\tau_{ij}(t) = \max(\tau_{ij}(t), \tau_{\min}) \tag{2.5}$$

where $\tau_{\min}$ is typically 0.01.

2.1.4 ACO Algorithm Flowchart

The complete ACO routing algorithm follows this iterative process:

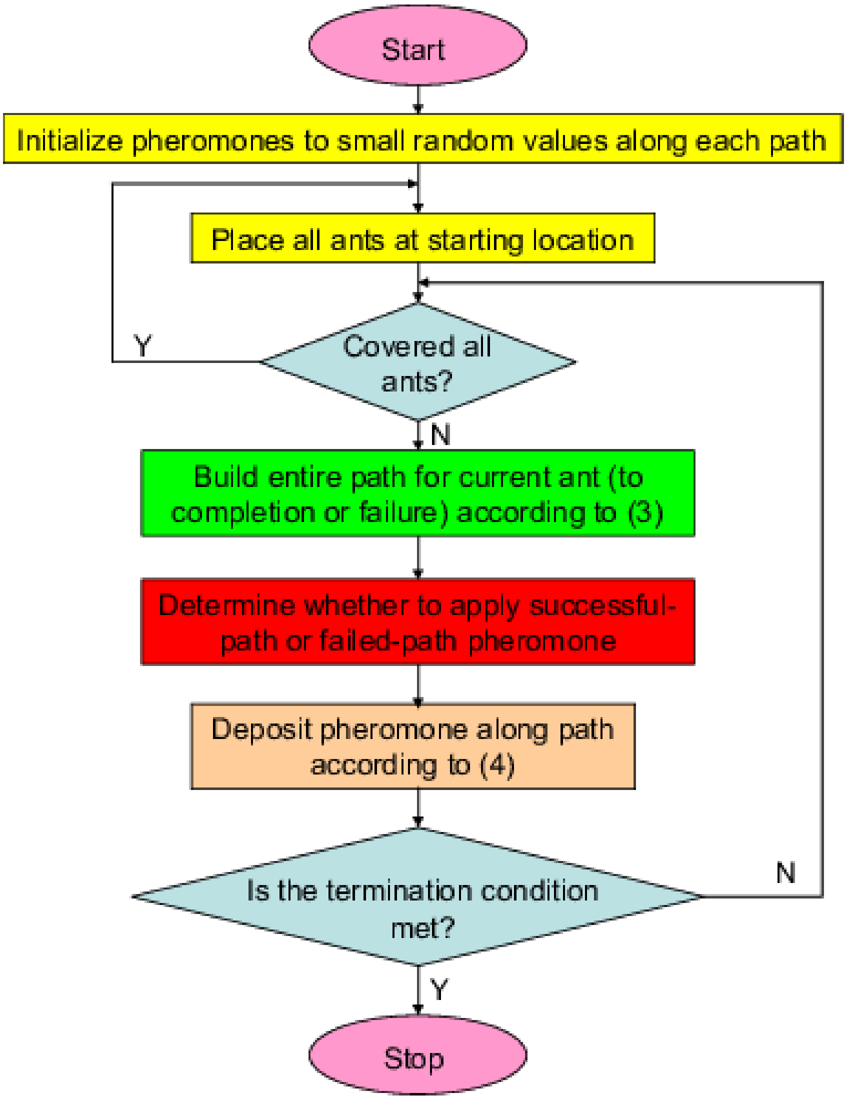

Figure 2.2: ACO Algorithm Flowchart for Network Routing

2.1.5 ACO Parameters

The behavior of ACO is controlled by several key parameters:

Parameter	Symbol	Description
Pheromone weight	α	Controls importance of pheromone trail (exploitation). Typical value: 1.0.
Visibility weight	β	Controls importance of link cost (exploration). Typical value: 2.0.
Evaporation rate	ρ	Fraction of pheromone that evaporates per iteration. Typical value: 0.1.
Deposit constant	Q	Controls amount of pheromone deposited. Typical value: 100.
Initial pheromone	τ_0	Starting pheromone on all links. Typical value: 1.0.
Number of ants	n	Ants launched per iteration. Typically equals number of routers.
Max iterations	$t_{\max}$	Number of learning cycles. Typical value: 5-10.

Table 2.1: ACO Algorithm Parameters

2.1.6 Pheromone Evolution Example

Consider a 3-router triangle network with link costs:

- Link $(0 \to 1)$: 25 ms
- Link $(0 \to 2)$: 10 ms
- Link $(2 \to 1)$: 5 ms

Initial State (Iteration 0):

$$\tau_{01} = \tau_{02} = \tau_{21} = 1.0$$

After Iteration 1: Ants discover that path $0 \to 2 \to 1$ (total cost 15 ms) is better than direct $0 \to 1$ (25 ms). Backward ants deposit:

$$\Delta\tau_{02} = \frac{100}{15} = 6.67, \quad \Delta\tau_{21} = \frac{100}{15} = 6.67$$

Updated pheromones:

$$\tau_{02} = 1.0 + 6.67 = 7.67, \quad \tau_{21} = 7.67, \quad \tau_{01} \approx 1.0$$

After Evaporation:

$$\tau_{02} = 0.9 \times 7.67 = 6.90, \quad \tau_{21} = 6.90, \quad \tau_{01} = 0.9$$

Over 5 iterations, short links accumulate high pheromone while expensive links lose pheromone, naturally guiding routing decisions toward optimal paths.

2.2 Network Routing Fundamentals

2.2.1 Routing Table

A routing table is a data structure stored in each router that maps destination addresses to next-hop routers. For a network with N routers, each router maintains $N - 1$ entries (one for each possible destination except itself).

Example Routing Table for Router 0:

Destination	Next Hop	Cost
1	2	15 ms
2	2	10 ms

Table 2.2: Routing Table Example

2.2.2 Link Cost Metrics

In AETHER, link cost represents transmission delay measured in milliseconds. The cost influences routing decisions through the visibility heuristic:

$$\eta_{ij} = \frac{1}{d_{ij}} \tag{2.6}$$

Lower cost links have higher visibility, making them more attractive to ants.

2.3 Congestion Awareness in Routing

Traditional ACO focuses solely on finding shortest paths, which can lead to congestion when all traffic converges on the same links. AETHER introduces **congestion-aware routing** by penalizing heavily-used links.

2.3.1 Link Usage Tracking

After constructing the routing table, AETHER counts how many routes use each link:

$$U_{ij} = |\{(s, d) : \text{Next hop from } s \text{ to } d \text{ is } j\}| \tag{2.7}$$

where U_{ij} is the usage count for link $(i \to j)$.

2.3.2 Congestion Penalty

Links with high usage receive a pheromone penalty to encourage load balancing:

$$\tau_{ij}^{\text{new}} = \tau_{ij} \times \text{penalty}(U_{ij}) \tag{2.8}$$

where:

$$\text{penalty}(U_{ij}) = \begin{cases} 1.0 & \text{if } U_{ij} = 1 \\ 0.8 & \text{if } U_{ij} = 2 \\ 0.6 & \text{if } U_{ij} = 3 \\ 0.4 & \text{if } U_{ij} \geq 4 \end{cases} \tag{2.9}$$

This mechanism distributes traffic across multiple paths, preventing network bottlenecks.

Chapter 3

Methodology

3.1 System Design and Architecture

AETHER is implemented in OMNeT++ simulator and consists of three main components working together to achieve adaptive routing:

1. **ACO Engine:** Manages ant generation, path exploration, and pheromone updates across multiple iterations.
2. **Routing Decision Module:** Implements two routing methods (Greedy and Probabilistic) to select next-hop routers.
3. **Packet Forwarding Engine:** Uses the learned routing table to forward data packets with fault tolerance.

The system operates in two phases:

Phase 1 - Offline Learning (Iterations 1-5): Ants explore the network and pheromone converges to optimal paths.

Phase 2 - Online Operation: Packets use the learned routing table for efficient forwarding.

3.2 Pheromone Table Calculation

3.2.1 Test Network Topology

The experiments use a **3-router triangle network** with the following characteristics:

- **3 Routers:** Router 0, Router 1, Router 2
- **3 Bidirectional Links:** Connected in a triangle topology
- **Link Costs:**

- Link ($0 \rightarrow 1$): 25 ms
- Link ($0 \rightarrow 2$): 10 ms
- Link ($1 \rightarrow 0$): 25 ms
- Link ($1 \rightarrow 2$): 5 ms
- Link ($2 \rightarrow 0$): 10 ms
- Link ($2 \rightarrow 1$): 5 ms

Figure 3.1 shows the network topology:

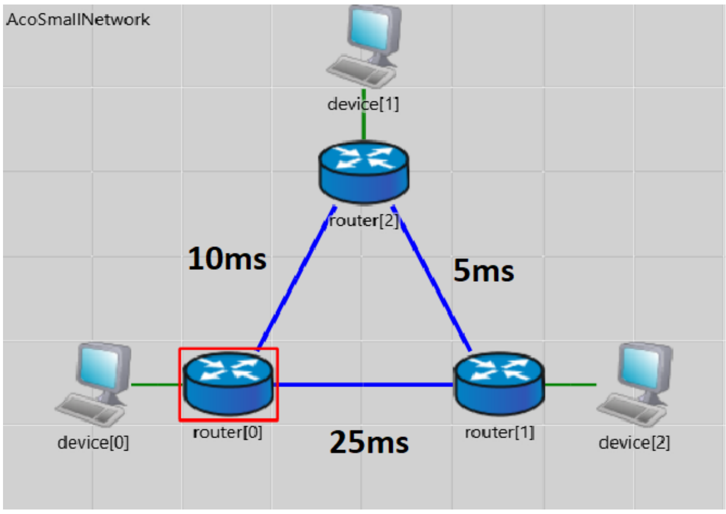

Figure 3.1: 3-Router Triangle Network Topology

3.2.2 Iteration 1 Calculation: Step-by-Step

Step 1: Initialize Pheromones

Before any ant exploration, all links are initialized with equal pheromone:

$$\tau_{ij}(0) = 1.0 \text{ for all links} \tag{3.1}$$

Initial Pheromone Table (Iteration 0):

Link	Pheromone	Cost (ms)	Visibility
$0 \rightarrow 1$	1.0	25	0.04
$0 \rightarrow 2$	1.0	10	0.10
$1 \rightarrow 0$	1.0	25	0.04
$1 \rightarrow 2$	1.0	5	0.20
$2 \rightarrow 0$	1.0	10	0.10
$2 \rightarrow 1$	1.0	5	0.20

Table 3.1: Initial Pheromone Table (Before Iteration 1)

where Visibility $\eta_{ij} = \frac{1}{\text{Cost}_{ij}}$.

Step 2: Launch 3 Forward Ants

Three ants are launched simultaneously from the three routers:

- **Ant 1:** From Router 0 to Router 1
- **Ant 2:** From Router 1 to Router 2
- **Ant 3:** From Router 2 to Router 0

Step 3: Ant Exploration

Each ant uses probabilistic selection to choose the next hop:

$$p_{ij} = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_k [\tau_{ik}]^\alpha \cdot [\eta_{ik}]^\beta} \tag{3.2}$$

with $\alpha = 1.0$ and $\beta = 2.0$.

Ant 1 (Router 0 to Router 1):

At Router 0, Ant 1 has two choices:

- **Option A:** Go directly to Router 1

$$\text{Score}_{0\to1} = (1.0)^{1.0} \times (0.04)^{2.0} = 1.0 \times 0.0016 = 0.0016$$

- **Option B:** Go to Router 2 first

$$\text{Score}_{0\to2} = (1.0)^{1.0} \times (0.10)^{2.0} = 1.0 \times 0.01 = 0.01$$

Total Score: $0.0016 + 0.01 = 0.0116$

Probabilities:

$$p_{0\to1} = \frac{0.0016}{0.0116} = 0.138 \quad (13.8\%)$$

$$p_{0\to2} = \frac{0.01}{0.0116} = 0.862 \quad (86.2\%)$$

Ant 1 probabilistically selects Router 2 (higher score). Path: $0 \to 2$, Cost: 10 ms.

At Router 2, Ant 1 must reach Router 1. Only option is:

$$p_{2\to1} = 1.0 \text{ (only unvisited neighbor)}$$

Ant 1 completes path: $0 \to 2 \to 1$, Total Cost: $10 + 5 = 15$ ms.

Ant 2 (Router 1 to Router 2):

At Router 1:

$$\text{Score}_{1\to0} = 1.0 \times 0.0016 = 0.0016$$

$$\text{Score}_{1\to 2} = 1.0 \times 0.04 = 0.04$$

Ant 2 selects Router 2 (direct path is better). Path: $1 \to 2$, Cost: 5 ms.

Ant 3 (Router 2 to Router 0):

At Router 2:

$$\text{Score}_{2\to 0} = 1.0 \times 0.01 = 0.01$$

$$\text{Score}_{2\to 1} = 1.0 \times 0.04 = 0.04$$

Ant 3 selects Router 1 (higher score). Path: $2 \to 1$, Cost: 5 ms.

At Router 1, Ant 3 reaches destination Router 0. Path: $2 \to 1 \to 0$, Total Cost: $5 + 25 = 30$ ms.

Step 4: Backward Ants Deposit Pheromone

Ants retrace their paths and deposit pheromone using:

$$\Delta\tau_{ij} = \frac{Q}{L_k} \tag{3.3}$$

with $Q = 100$.

Ant 1 Deposits on Path 0→2→1 (Total Cost: 15 ms):

$$\Delta\tau_{0\to 2} = \frac{100}{15} = 6.67, \quad \Delta\tau_{2\to 1} = 6.67$$

Ant 2 Deposits on Path 1→2 (Total Cost: 5 ms):

$$\Delta\tau_{1\to 2} = \frac{100}{5} = 20.0$$

Ant 3 Deposits on Path 2→1→0 (Total Cost: 30 ms):

$$\Delta\tau_{2\to 1} = \frac{100}{30} = 3.33, \quad \Delta\tau_{1\to 0} = 3.33$$

Step 5: Update Pheromone Table

Apply pheromone updates:

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \Delta\tau_{ij} \tag{3.4}$$

After All Ant Deposits:

Step 6: Evaporate Pheromones

Apply evaporation with $\rho = 0.1$ (10% evaporation):

Link	Before	Deposit	After	Source
$0 \to 1$	1.0	0.0	1.0	—
$0 \to 2$	1.0	6.67	7.67	Ant 1
$1 \to 0$	1.0	3.33	4.33	Ant 3
$1 \to 2$	1.0	20.0	21.0	Ant 2
$2 \to 0$	1.0	0.0	1.0	—
$2 \to 1$	1.0	6.67 + 3.33	11.0	Ant 1, Ant 3

Table 3.2: Pheromone After Deposits (Before Evaporation)

$$\tau_{ij}(t+1) = (1-0.1) \times \tau_{ij}(t) = 0.9 \times \tau_{ij}(t) \tag{3.5}$$

After Evaporation:

Link	Before Evap	After Evap
$0 \to 1$	1.0	0.90
$0 \to 2$	7.67	6.90
$1 \to 0$	4.33	3.90
$1 \to 2$	21.0	18.90
$2 \to 0$	1.0	0.90
$2 \to 1$	11.0	9.90

Table 3.3: Pheromone After Evaporation (End of Iteration 1)

Step 7: Final Pheromone Table (Iteration 1)

Link	Pheromone	Visibility	Score
$0 \to 1$	0.90	0.04	0.0014
$0 \to 2$	6.90	0.10	0.069
$1 \to 0$	3.90	0.04	0.0062
$1 \to 2$	18.90	0.20	0.756
$2 \to 0$	0.90	0.10	0.009
$2 \to 1$	9.90	0.20	0.396

Table 3.4: Pheromone Table After Iteration 1 (Final)

This pattern repeats for Iterations 2-5, with pheromone progressively converging to the best paths identified by the ants.

3.3 Routing Table Construction

3.3.1 Generalized Routing Table Construction Algorithm

Overview: After ACO learning completes, a routing table is constructed that maps every (source, destination) pair to an optimal next-hop router.

Algorithm Input:

- Pheromone table τ_{ij} learned from $t_{\max}$ ACO iterations
- Network topology (routers and links)
- Visibility values $\eta_{ij} = \frac{1}{d_{ij}}$

Algorithm Output:

- Centralized routing table: $\text{RT}(s, d) \rightarrow n$
- Load distribution report

Routing Table Construction Steps:

1. **Initialize:** Create empty routing table RT and link usage counter $U_{ij} = 0$
2. **For each source router** $s \in [0, N-1]$:
   (a) For each destination router $d \in [0, N-1]$ where $s \neq d$:
      i. Get all neighbors of router s: $\mathcal{N}_s$
      ii. **If GREEDY Method:**

$$\text{next\_hop} = \arg\max_{n \in \mathcal{N}_s} [\tau_{sn}]^\alpha \times [\eta_{sn}]^\beta \tag{3.6}$$

      iii. **If PROBABILISTIC Method:**
         A. Calculate score for each neighbor: $p_n = [\tau_{sn}]^\alpha \times [\eta_{sn}]^\beta$
         B. Normalize: $p_n \leftarrow \frac{p_n}{\sum_k p_k}$
         C. Roulette wheel selection: Generate random $r \in [0, 1]$, select neighbor where cumulative probability exceeds r
      iv. Store decision: $\text{RT}(s, d) \leftarrow \text{next\_hop}$
      v. Count link usage: $U_{s,\text{next\_hop}} \leftarrow U_{s,\text{next\_hop}} + 1$
3. **Apply Congestion Penalty:**

(a) For each link (i, j) with usage count U_{ij}:

$$\text{penalty}(U_{ij}) = \begin{cases} 1.0 & \text{if } U_{ij} = 1 \\ 0.8 & \text{if } U_{ij} = 2 \\ 0.6 & \text{if } U_{ij} = 3 \\ 0.4 & \text{if } U_{ij} \geq 4 \end{cases} \tag{3.7}$$

(b) Update pheromone: $\tau_{ij} \leftarrow \tau_{ij} \times \text{penalty}(U_{ij})$

(c) Re-run routing table construction with penalized pheromones

4. **Return:** Finalized routing table RT

Algorithm Complexity:
The algorithm has time complexity $\mathcal{O}(N^3)$:

- $N(N-1) \approx \mathcal{O}(N^2)$ source-destination pairs
- For each pair, check $\mathcal{O}(N)$ neighbors
- Total: $\mathcal{O}(N^3)$

For a 3-router network: $3 \times 2 \times 3 = 18$ operations, completing instantly.

3.3.2 Routing Table Construction Using Learned Pheromones

After 5 ACO iterations, the pheromone table has converged to the values shown in Table 3.5. We now construct the routing table using both methods with detailed step-by-step calculations.

Final Pheromone Table (After Iteration 5)

Link	Pheromone τ	Cost (ms)	Visibility η	Score
$0 \to 1$	0.59	25	0.04	0.0009
$0 \to 2$	49.73	10	0.10	0.4973
$1 \to 0$	25.16	25	0.04	0.0403
$1 \to 2$	74.30	5	0.20	2.9721
$2 \to 0$	0.59	10	0.10	0.0059
$2 \to 1$	74.30	5	0.20	2.9721

Table 3.5: Final Pheromone Table (Iteration 5)

where Score $= \tau^\alpha \times \eta^\beta$ with $\alpha = 1.0$ and $\beta = 2.0$.

3.3.3 Method 1: Greedy Local - Step-by-Step Calculation

Route 1: Router 0 to Router 1

From Router 0, evaluate all neighbors:

Neighbor A: Router 1 (Direct)

$$\begin{aligned}
\text{Score}_{0\to1} &= [\tau_{0\to1}]^{\alpha} \times [\eta_{0\to1}]^{\beta} \\
&= (0.59)^{1.0} \times (0.04)^{2.0} \\
&= 0.59 \times 0.0016 \\
&= 0.0009
\end{aligned}$$

Neighbor B: Router 2

$$\begin{aligned}
\text{Score}_{0\to2} &= (49.73)^{1.0} \times (0.10)^{2.0} \\
&= 49.73 \times 0.01 \\
&= 0.4973
\end{aligned}$$

Decision: $\max(0.0009, 0.4973) = 0.4973 \to$ **Select Router 2**

Route 0 → 1: Next Hop = 2

Route 2: Router 0 to Router 2

Same calculations as Route 1 (destination doesn't affect local greedy choice):

Decision: Select Router 2 (direct)

Route 0 → 2: Next Hop = 2

Route 3: Router 1 to Router 0

From Router 1, evaluate neighbors:

Neighbor A: Router 0 (Direct)

$$\begin{aligned}
\text{Score}_{1\to0} &= (25.16)^{1.0} \times (0.04)^{2.0} \\
&= 25.16 \times 0.0016 \\
&= 0.0403
\end{aligned}$$

Neighbor B: Router 2

$$\begin{aligned}
\text{Score}_{1\to2} &= (74.30)^{1.0} \times (0.20)^{2.0} \\
&= 74.30 \times 0.04 \\
&= 2.9721
\end{aligned}$$

Decision: $\max(0.0403, 2.9721) = 2.9721 \rightarrow$ **Select Router 2**

Route 1 → 0: Next Hop = 2

Route 4: Router 1 to Router 2

Same evaluation from Router 1:

Decision: Select Router 2 (direct)

Route 1 → 2: Next Hop = 2

Route 5: Router 2 to Router 0

From Router 2, evaluate neighbors:

Neighbor A: Router 0 (Direct)

$$\begin{aligned}\text{Score}_{2\to 0} &= (0.59)^{1.0} \times (0.10)^{2.0} \\ &= 0.59 \times 0.01 \\ &= 0.0059\end{aligned}$$

Neighbor B: Router 1

$$\begin{aligned}\text{Score}_{2\to 1} &= (74.30)^{1.0} \times (0.20)^{2.0} \\ &= 74.30 \times 0.04 \\ &= 2.9721\end{aligned}$$

Decision: $\max(0.0059, 2.9721) = 2.9721 \rightarrow$ **Select Router 1**

Route 2 → 0: Next Hop = 1

Route 6: Router 2 to Router 1

Same evaluation from Router 2:

Decision: Select Router 1 (direct)

Route 2 → 1: Next Hop = 1

Greedy Local Routing Table Summary:

Source	Dest	Next Hop	Winning Score	Path
0	1	2	0.4973	0→2→1
0	2	2	0.4973	0→2
1	0	2	2.9721	1→2→0
1	2	2	2.9721	1→2
2	0	1	2.9721	2→1→0
2	1	1	2.9721	2→1

Table 3.6: Method 1: Greedy Local Routing Table

Link Usage Count

Count how many routes use each link:

Link	Routes Using Link	Usage Count U
$0 \to 2$	0→1, 0→2	2
$1 \to 2$	1→0, 1→2	2
$2 \to 1$	2→0, 2→1	2
$0 \to 1$	None	0
$1 \to 0$	None	0
$2 \to 0$	None	0

Table 3.7: Link Usage Count from Greedy Routing

3.3.4 Method 2: Probabilistic with Congestion Awareness

Step 1: Apply Congestion Penalty to Pheromones

For links with usage count $U \geq 2$, apply 20% penalty:

$$\tau_{0\to2}^{\text{new}} = 49.73 \times 0.8 = 39.78$$
$$\tau_{1\to2}^{\text{new}} = 74.30 \times 0.8 = 59.44$$
$$\tau_{2\to1}^{\text{new}} = 74.30 \times 0.8 = 59.44$$

Updated Pheromone Table After Penalty:

Link	Original τ	Penalty	New τ	New Score
$0 \to 1$	0.59	1.0	0.59	0.0009
$0 \to 2$	49.73	0.8	39.78	0.3978
$1 \to 0$	25.16	1.0	25.16	0.0403
$1 \to 2$	74.30	0.8	59.44	2.3776
$2 \to 0$	0.59	1.0	0.59	0.0059
$2 \to 1$	74.30	0.8	59.44	2.3776

Table 3.8: Pheromone After Congestion Penalty

Step 2: Probabilistic Selection - Route by Route

Route 1: Router 0 to Router 1

Calculate neighbor probabilities:

$$\text{Score}_{0\to1} = 0.0009$$
$$\text{Score}_{0\to2} = 0.3978$$
$$\text{Total} = 0.3987$$

Normalized probabilities:

$$P_{0\to 1} = \frac{0.0009}{0.3987} = 0.0023 = 0.23\%$$
$$P_{0\to 2} = \frac{0.3978}{0.3987} = 0.9977 = 99.77\%$$

Roulette Wheel:

- Slot [0.0000, 0.0023): Router 1
- Slot [0.0023, 1.0000]: Router 2

Random value: $r = 0.654 \to$ Falls in Router 2 slot

Decision: Route 0 → 1: Next Hop = 2

Route 3: Router 1 to Router 0

$$\text{Score}_{1\to 0} = 0.0403$$
$$\text{Score}_{1\to 2} = 2.3776$$
$$\text{Total} = 2.4179$$

Normalized probabilities:

$$P_{1\to 0} = \frac{0.0403}{2.4179} = 0.0167 = 1.67\%$$
$$P_{1\to 2} = \frac{2.3776}{2.4179} = 0.9833 = 98.33\%$$

Random value: $r = 0.342 \to$ Router 2

Decision: Route 1 → 0: Next Hop = 2

Route 5: Router 2 to Router 0

$$\text{Score}_{2\to 0} = 0.0059$$
$$\text{Score}_{2\to 1} = 2.3776$$
$$\text{Total} = 2.3835$$

Normalized probabilities:

$$P_{2\to0} = \frac{0.0059}{2.3835} = 0.0025 = 0.25\%$$
$$P_{2\to1} = \frac{2.3776}{2.3835} = 0.9975 = 99.75\%$$

Random value: $r = 0.891 \to$ Router 1

Decision: Route 2 → 0: Next Hop = 1

Probabilistic Routing Table Summary:

Source	Dest	Next Hop	Selection Probability
0	1	2	99.77%
0	2	2	99.77%
1	0	2	98.33%
1	2	2	98.33%
2	0	1	99.75%
2	1	1	99.75%

Table 3.9: Method 2: Probabilistic Routing Table

Comparison of Both Methods

Route	Greedy	Probabilistic	Match	Path Cost
0→1	2	2	Yes	15 ms
0→2	2	2	Yes	10 ms
1→0	2	2	Yes	15 ms
1→2	2	2	Yes	5 ms
2→0	1	1	Yes	30 ms
2→1	1	1	Yes	5 ms

Table 3.10: Comparison of Greedy vs Probabilistic Routing

Observation: Both methods produce identical routing tables because pheromone differences are so large (74.30 vs 0.59) that even with 20% congestion penalty, the probabilistic selection still heavily favors the same neighbors as greedy selection.

Chapter 4

Discussion and Conclusion

4.1 Discussion

This project shows that using Ant Colony Optimization (ACO) allows a network to automatically discover and reinforce good paths while adapting to varying traffic and failures. Even in a simple 3-router triangle, the system quickly learned to prefer faster, multi-hop routes and balance load, with both Greedy and Probabilistic methods achieving mostly optimal routing. The main limitation of the presented results is the small, static test topology, which restricts the demonstration of ACO's full capabilities.

4.2 Conclusion and Future Work

AETHER proves that adaptive, bio-inspired routing can provide robust, fault-tolerant, and efficient routing without depending on static tables or single-path decisions. For further study, this work should be extended by applying ACO to larger and more dynamic networks, supporting multiple routing criteria like bandwidth or reliability, developing true online learning where ants and packets move together, and directly comparing its performance with established protocols such as OSPF and BGP.

The End