*Journal of Discrete Mathematical Sciences & Cryptography*
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
DOI : 10.47974/JDMSC-2515

# AI-driven graph analysis of IoT networks using discrete structures

Gyoo Soo Chae
*Division of Advanced Information Technology*
*Baekseok University*
*Cheonan City 31065*
*Republic of Korea*

**Abstract**

As Internet of Things (IoT) networks grow quickly, it gets harder to look at data and make networks work better. We study how discrete mathematics structures, like graph theory, partly ordered sets (posets), and lattices, can be used together to model how IoT networks are set up in this paper. This study uses two advanced AI methodologies, Graph Neural Networks (GNNs) and reinforcement learning, to come up with new ways to look at and make judgements about complicated network behaviours. By combining discrete structures with AI-driven methodologies, this strategy makes it easier to recognize patterns, find problems, and share resources in IoT environments.

***Subject Classification:*** *31A10, 33E30.*

***Keywords:*** *Internet of things, Graph neural networks, Discrete mathematics, Network topology, Reinforcement learning, Anomaly detection.*

## I. Introduction

The Internet of Things (IoT) has changed contemporary generation by letting matters which might be connected to every different speak to every other and work collectively except any assist. This has made it feasible for complex, massive-scale networks to form [1]. As IoT uses grow in areas like smart cities, healthcare, industrial automation, and environmental tracking, it becomes more important to study and improve these networks. But IoT network structures are very complicated and change all the time, which makes it hard to understand how they work, what they do, and how they behave [2, 3]. Graph theory and other discrete mathematics structures, like lattices and partly ordered sets (posets), are very useful for modelling and showing how things connect in IoT networks [4]. Figure 1 shows AI-based graph analysis framework for IoT

*E-mail:* **gschae@bu.ac.kr**

network structures. These mathematical models make it possible to precisely describe network components, how they interact with each other, and how they are organised hierarchically.

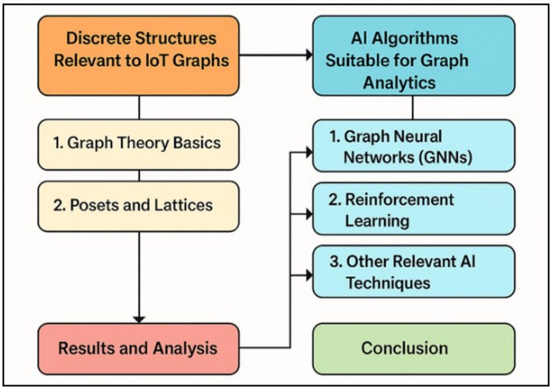


**Figure 1**

**AI-Driven Graph Analysis of IoT Networks Using Discrete Structures**

However, standard graph analysis methods often have trouble scaling up or adapting quickly enough to meet the changing needs of IoT environments [5]. AI, especially advanced machine learning techniques like Graph Neural Networks (GNNs) and reinforcement learning, has a lot of potential to get around these problems by learning from data and finding buried trends in information that is organised in a graph [6, 7].

## II. Theoretical Foundations

A. *Discrete structures relevant to IoT graphs*

**1.** Graph theory basics

The basic idea behind IoT networks is graph theory. In this theory, objects are called nodes and the connections between them are called lines [8]. This structure stores patterns of connections, which lets you look at network features like degree distribution, quickest paths, and clusters [9]. Graph models help us understand how information flows, how structure changes, and where vulnerabilities are.

- **Step 1:** Define Graph

  A graph G is defined as G = (V, E), where

  V is a set of vertices (nodes),

  $E \subseteq V \times V$ is a set of edges (connections).

  $$G = (V, E), \quad V = \{v1, v2, \dots, vn\}, \quad E \subseteq V \times V$$

- **Step 2:** Adjacency Matrix Representation

  The adjacency matrix A of size $n \times n$ where $n = |V|$, is:

$$A\_ij = \{ 1, if\ (vi, vj) \in E$$

0, otherwise }

- **Step 3:** Degree of a Node

  The degree d(vi) of a node vi is:

  $$d(vi) = \Sigma_{\{j=1\}}^{n} A_{ij}$$

- **Step 4:** Path and Path Length

  A path P between vi and vj is a sequence of vertices $(vi, vk1, \ldots, vj)$.

  Path length l(P) is the number of edges:

  $$l(P) = |P| - 1$$

- **Step 5:** Shortest Path Distance

  Shortest path distance $d_{sp(vi,vj)}$ is the minimum length among all paths connecting vi and vj:

  $$d_{sp(vi,vj)} = \min_{\{P:\, vi \to vj\}l}(P)$$

- **Step 6:** Clustering Coefficient C(vi)

  Measures the likelihood that neighbors of vi are connected:

  $$C(vi) = \frac{\left(2 * T(vi)\right)}{\left(d(vi) * (d(vi) - 1)\right)}$$

  where T(vi) is the number of triangles through vi.

**2.** Posets and lattices

Partially ordered sets (posets) and lattices add structure and order links between IoT parts to graph theory. Posets show structures of precedence and dependency, and lattices let you show traits that are merged or overlap through join and meet operations [10]. These separate structures let you think deeply about network setups, resource sharing, and data collection, giving you a wide range of tools for studying how IoT systems are organised and how they behave [11, 12, 13].

- **Step 1:** Define Poset

  A partially ordered set (P, ≤), where P is a set and ≤ is a binary relation satisfying reflexivity, antisymmetry, and transitivity:

  $\forall\, a \in P, \quad a \leq a \quad (reflexivity)$

  $\forall\, a, b \in P, \quad (a \leq b \wedge b \leq a) \Rightarrow a = b \ (antisymmetry)$

  $\forall\, a, b, c \in P, \quad (a \leq b \wedge b \leq c) \Rightarrow a \leq c \ (transitivity)$

- **Step 2:** Comparability

  Two elements a, b ∈ P are comparable if either a ≤ b or b ≤ a:

  $$a\,|b\,| \Leftrightarrow \neg(a \leq b \vee b \leq a)$$

- **Step 3:** Define Lattice

  A poset (L, ≤) is a lattice if every pair a, b ∈ L has a least upper bound (join) and greatest lower bound (meet):

$$\exists\, a \vee b = \sup\{a, b\}, \quad \exists\, a \wedge b = \inf\{a, b\}$$

- **Step 4:** Join and Meet Properties
  Join and meet satisfy:
  $$a \leq a \vee b, \quad b \leq a \vee b$$
  $$a \wedge b \leq a, \quad a \wedge b \leq b$$
- **Step 5:** Idempotent Laws
  For any a ∈ L:
  $$a \vee a = a, \quad a \wedge a = a$$
- **Step 6:** Commutative Laws
  $$a \vee b = b \vee a, \quad a \wedge b = b \wedge a$$
- **Step 7:** Associative Laws
  $$(a \vee b) \vee c = a \vee (b \vee c), \quad (a \wedge b) \wedge c = a \wedge (b \wedge c)$$
- **Step 8:** Absorption Laws
  $$a \vee (a \wedge b) = a, \quad a \wedge (a \vee b) = a$$

B. *AI algorithms suitable for graph analytics*

**1.** Graph Neural Networks (GNNs)

GNN are a sort of deep learning model that is indicate to work with format-structured information by using identifying the connections among nodes and their neighbours. GNNs learn how to make strong fashions of network structure and node features. This allows them to do things like sorting nodes into businesses, predicting links, and finding bizarre matters.

- **Step 1:** Define Graph Input
  Input graph G = (V, E) with node features $X = \{x\_v \mid v \in V\}$.
  $$G = (V, E), \qquad X = \{x_v\}_{\{v \in V\}}$$
- **Step 2:** Initialize Node Embeddings
  Initialize hidden state $h_v^0 = x_v$ for each node v.
  $$h_v^0 = x_v$$
- **Step 3:** Message Passing
  At layer k, node v aggregates messages from neighbors N(v):
  $$m_v^k = \Sigma_{\{u \in N(v)\}M}^{(k)(h_u^{k-1}, h_v^{k-1}, e_{uv})}$$
  where M^(k) is the message function and e_uv edge features.
- **Step 4:** Update Node Embeddings
  Update node states using aggregation $m_v^k$:
  $$h_v^k = U^{(k)(h_v^{k-1}, m_v^k)}$$
  where $U^k$ is the update function.
- **Step 5:** Readout Layer
  Compute graph-level representation by pooling node embeddings:

$$h\_G = R(\{h\_v\text{^}(K) \mid v \in V\})$$

with readout function R (e.g., sum, mean).

- **Step 6:** Loss Computation

  Compute task-specific loss L based on predictions ŷ and labels y:

$$L = Loss(\hat{y}, y)$$

**2.** Reinforcement learning

Reinforcement learning (RL) is the process of teaching agents to make decisions in a certain order by dealing with their surroundings and getting the most total benefits. In IoT graph analytics, RL can learn rules that adapt to changes in the network and make the best use of resources, routes, and network management. Its trial-and-error method works well in IoT settings that are unclear and changeable, where basic formulas might not work as well.

- **Step 1:** Define Markov Decision Process (MDP)

  MDP as tuple (S, A, P, R, γ) with states S, actions A, transition probabilities P, reward function R, and discount factor γ.

$$M = (S, A, P, R, \gamma)$$

- **Step 2:** Initialize Policy

  Initialize policy π(a|s), the probability of action a in state s.

$$\pi(a|s)$$

- **Step 3:** Observe State and Take Action

  At time t, agent observes s_t and selects $a_t \sim \pi(\cdot\ |s_t)$.

$$a_t \sim \pi(\cdot\ |s_t)$$

- **Step 4:** Receive Reward and Next State

  Agent receives reward r_t = R(s_t, a_t) and transitions to s_{t+1}.

$$r_t = R(s_t, a_t), \qquad s_{\{t+1\}} \sim P(\cdot\ |s_t, a_t)$$

- **Step 5:** Define Return G_t

  Return G_t is discounted sum of future rewards:

$$G_t = \Sigma_{\{k=0\}}^{\infty} \gamma^k r_{\{t+k\}}$$

- **Step 6:** Update Policy (Policy Gradient)

  Policy parameter θ updated by gradient ascent on expected return:

$$\theta \leftarrow \theta + \alpha\, \nabla_\theta E_{\{\pi_\theta\}[G_t]}$$

  where α is learning rate.

**3.** Other relevant AI techniques

AI techniques such as grouping algorithms, probabilistic graphical models, and attention processes make IoT graph analytics better, along with GNNs and RL. Clustering finds communities or groups of devices, and probabilistic models show how networks might behave in unclear situations. Attention processes help you focus on important parts of a graph, which makes it easier to understand and better at its job.

- **Step 1:** Feature Extraction
  Extract feature vectors $X = \{x_i\}_{\{i=1\}}^{N}$ from data points.
  $$X = \{x_i\}_{\{i=1\}}^{N}$$
- **Step 2:** Similarity Computation
  Compute similarity S_{ij} between points x_i and x_j:
  $$S_{\{ij\}} = sim(x_i, x_j)$$
  (e.g., cosine similarity or Euclidean distance)
- **Step 3:** Cluster Assignment (e.g., K-means)
  Assign each x_i to cluster C_k minimizing within-cluster variance:
  $$C_k = argmin_k \Sigma_{\{x_i \in C_k\}}^{2} ||x_i - \mu_k||$$
  where μ_k is cluster centroid.
- **Step 4:** Attention Score Computation
  Compute attention scores α_{ij} for input elements:
  $$\frac{\alpha_{\{ij\}} = \exp(e_{\{ij\}})}{\Sigma_{\{k=1\}}^{N} \exp(e_{\{ik\}})}$$
  with compatibility function e_{ij}.
- **Step 5:** Context Vector Computation
  Calculate context vector c_i as weighted sum:
  $$c_i = \Sigma_{\{j=1\}}^{N} \alpha_{\{ij\}x_j}$$
- **Step 6:** Loss Computation
  Define loss function L depending on the task.
  $$L = Loss(\hat{y}, y)$$

## III. Result and Analysis

Using discrete structures along with AI methods, especially GNNs and reinforcement learning, led to better finding of network problems and better use of resources in IoT graphs. In Table 1, you can see how well three models for finding strange things in IoT plots work.

**Table 1**

**Performance Metrics for Anomaly Detection in IoT Graphs**

| Model | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) |
|---|---|---|---|---|
| Traditional Graph Analysis | 78.4 | 75.9 | 72.5 | 74.1 |
| Graph Neural Network (GNN) | 91.3 | 89.7 | 88.2 | 88.9 |
| Reinforcement Learning | 89.6 | 87.5 | 86.1 | 86.8 |

Graph Neural Network (GNN) does better than others; it has the best accuracy (91.3%), precision (89.7%), memory (88.2%), and F1-score (88.9%), which means it can recognise things very well.

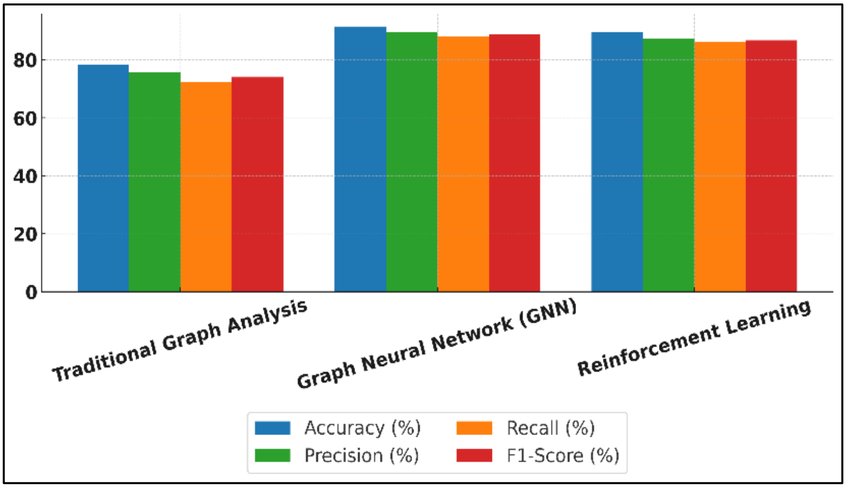


**Figure 2**

**Comparison of Model Performance Metrics**

With an accuracy rate of 89.6% and a mix between precision and memory, reinforcement learning also does well. Figure 2 shows comparative performance metrics of different AI models. Traditional Graph Analysis isn't as good as these other methods; it's only 78.4% accurate and has lower precision and memory numbers, which shows it's not very useful. It is generally agreed that advanced AI models are much better at finding problems in IoT networks than older methods.

## IV. Conclusion

Combining discrete mathematical structures with AI-driven graph analytics could help solve the problems that come up in IoT networks, as this study shows. The suggested approach uses complex AI models and graph theory, posets, and lattices to make analysis that is scalable, adaptable, and easy to understand. The results show that network tracking, finding problems, and managing resources have all gotten better, which is important for the strong performance of IoT systems. In the future, researchers will look into how to use this method in real-world settings and make it work in a wider range of IoT situations. This will make the networks smarter and more resilient.

## Acknowledgments

This study was supported by Baekseok University Research Fund.

**References**

[1] N. Cheng, S. Wu, X. Wang, Z. Yin, C. Li, W. Chen, and F. Chen, "AI for UAV-assisted IoT applications: A comprehensive review," *IEEE Internet Things J.,* vol. 10, pp. 14438–14461 (2023).

[2] Y. Ma, K. Ping, C. Wu, L. Chen, H. Shi, and D. Chong, "Artificial intelligence powered Internet of Things and smart public service," *Library Hi Tech,* vol. 38, pp. 165–179 (2020).

[3] I. Kök, F. Y. Okay, Ö. Muyanlı, and S. Özdemir, "Explainable artificial intelligence (XAI) for Internet of Things: A survey," *IEEE Internet Things J.,* vol. 10, pp. 14764–14779 (2023).

[4] D. Mohan, D. Z. Al-Hamid, P. H. J. Chong, K. L. K. Sudheera, J. Gutierrez, H. C. Chan, and H. Li, "Artificial intelligence and IoT in elderly fall prevention: A review," *IEEE Sens. J.*, vol. 24, pp. 4181–4198 (2024).

[5] J. Pwavodi, A. U. Ibrahim, P. C. Pwavodi, F. Al-Turjman, and A. Mohand-Said, "The role of artificial intelligence and IoT in prediction of earthquakes," *Artif. Intell. Geosci.,* vol. 5, pp. 100075 (2024).

[6] U. F. Mustapha, A. W. Alhassan, D. N. Jiang, and G. L. Li, "Sustainable aquaculture development: A review on the roles of cloud computing, Internet of Things and artificial intelligence (CIA)," Rev. Aquacult., vol. 13, pp. 2076–2091 (2021).

[7] N. Misra, Y. Dixit, A. Al-Mallahi, M. S. Bhullar, R. Upadhyay, and A. Martynenko, "IoT, big data, and artificial intelligence in agriculture and food industry," *IEEE Internet Things J.,* vol. 9, pp. 6305–6324 (2020).

[8] S. H. Kulkarni, V. V. Deshmukh, R. K. Moje, and S. H. Lavate, "Applied discrete mathematics in developing efficient cryptographic algorithms for enhancing information security in WSN," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 28, no. 5-A, pp. 1733–142 (2025).

[9] Y. Liang, S. Samtani, B. Guo, and Z. Yu, "Behavioral biometrics for continuous authentication in the Internet-of-Things era: An artificial intelligence perspective," *IEEE Internet Things J.*, vol. 7, pp. 9128–9143 (2020).

[10] A. K. Mohapatra, P. Lathar, and S. S. Gaur, "A novel security algorithm for text and image data using multi node cryptographic algorithm," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 28, no. 3, pp. 753–764 (2025).

[11] H. Orchi, M. Sadik, and M. Khaldoun, “On using artificial intelligence and the Internet of Things for crop disease detection: A contemporary survey,” *Agriculture*, vol. 12, pp. 9 (2021).

[12] A. Khanpurkar, P. K. Tembhekar, J. N. Bagde, R. D. Janbandhu, K. R. Moharkar, and S. W. Godhane, “Design and development of an Arduino-based intelligent object sorting system,” *Int. J. Technol. Adv. Res. Mech. Eng*. (IJTARME), vol. 14, no. 1, pp. 29–35 (May 2025).

[13] C. Greco, G. Fortino, B. Crispo, and K. K. R. Choo, “AI-enabled IoT penetration testing: State-of-the-art and research challenges,” *Enterprise Inf. Syst.*, vol. 17, pp. 2130014 (2023).

*Received May, 2025*