C:\Users\janani\Desktop\RGU.PNG

BSc (Hons) Artificial Intelligence and Data Science

**Module: CM2602 - Artificial Intelligence**

Individual Coursework Report

**Module Leader: Nipuna Senanayake**

RGU Student ID : 2410213

IIT Student ID : 20233168

Student Name : LOGANATHAN THUSHARKANTH

Contents

[***QUESTION 1 : Shift Allocation Optimization Using Excel Solver*** 1](#_Toc196750660)

[**1.** **Introduction** 1](#_Toc196750661)

[**2.** **Problem Description** 1](#_Toc196750662)

[**3.** **Variable Setup** 1](#_Toc196750663)

[**4.** **Solver Constraints** 2](#_Toc196750664)

[**5.** **Objective Function** 2](#_Toc196750665)

[**6.** **Preferences (Soft Constraints)** 2](#_Toc196750666)

[**7.** **Solver Implementation Screenshots** 3](#_Toc196750667)

[**8.** **Results & Analysis** 4](#_Toc196750668)

[**9.** **Discussion** 5](#_Toc196750669)

[**10.** **Conclusion** 6](#_Toc196750670)

[**QUESTION 2 : CYBERSECURITY ONTOLOGY DEVELOPMENT** 7](#_Toc196750671)

[**Introduction** 7](#_Toc196750672)

[**a.** **Define the scope of the ontology:** 7](#_Toc196750673)

[**b.** **Build Concept Graph (Taxonomy)** 8](#_Toc196750674)

[**c.** **Create OWL/RDF Ontology** 10](#_Toc196750675)

[**d.** **SPARQL Queries** 14](#_Toc196750676)

[**QUESTION 03– UCS & A\* IN 6x6 MAZE** 15](#_Toc196750677)

[**6.** **Introduction** 15](#_Toc196750678)

[**1.** **Task 1** 15](#_Toc196750679)

[**a.** **Coordinate Representation of Nodes** 15](#_Toc196750680)

[**b.** **Random Start Node (0 to 11)** 15](#_Toc196750681)

[**c.** **Random Goal Node (24 to 35)** 16](#_Toc196750682)

[**d.** **Random Barrier Nodes** 16](#_Toc196750683)

[**Task 2** 17](#_Toc196750684)

[**a.** **Process neighbors in increasing order** 17](#_Toc196750685)

[**b.** **Only valid moves (horizontal, vertical, diagonal)** 17](#_Toc196750686)

[**c.** **No moves through barrier nodes** 17](#_Toc196750687)

[**d.** **Time to find the goal** 17](#_Toc196750688)

[**e.** **Edge cost using Chebyshev distance** 18](#_Toc196750689)

[**f.** **Final Output Evidence:** 18](#_Toc196750690)

[**Task 3: Heuristic Function** 19](#_Toc196750691)

[**Task 4: A\* Search** 20](#_Toc196750692)

[ **Import and Chebyshev Distance:** 20](#_Toc196750693)

[ **Main A\* Function:** 20](#_Toc196750694)

[ **Helper: Reconstruct Path:** 21](#_Toc196750695)

[ **Helper: Get Neighbors:** 22](#_Toc196750696)

[ **Example Test Case:** 22](#_Toc196750697)

[ **Final output** 23](#_Toc196750698)

[**Task 5: Run 3 Random Mazes** 23](#_Toc196750699)

[ Experiment Results 23](#_Toc196750700)

[ Analysis of Results 24](#_Toc196750701)

[ Statistical Analysis 25](#_Toc196750702)

[ Observations 25](#_Toc196750703)

[ Final outputs 26](#_Toc196750704)

[**Conclusion** 26](#_Toc196750705)

[**QUESTION 4 - Fuzzy Logic-Based Anomaly Detection and Correction in Smart Grid Systems** 27](#_Toc196750706)

[**1.** **Introduction** 27](#_Toc196750707)

[**2.** **Anomaly Detection** 27](#_Toc196750708)

[ **Voltage Membership Function** 27](#_Toc196750709)

[ **Frequency Membership Function** 28](#_Toc196750710)

[ **Load Imbalance Membership Function** 28](#_Toc196750711)

[**3.** **Fault Mitigation** 29](#_Toc196750712)

[ **Defuzzification (Converting Fuzzy Outputs into Severity Score)** 29](#_Toc196750713)

[ **Deciding Corrective Actions Based on Severity** 30](#_Toc196750714)

[**4.** **Fuzzification** 31](#_Toc196750715)

[ **Membership function for Voltage Deviation** 31](#_Toc196750716)

[ **Membership function for Frequency Variation** 31](#_Toc196750717)

[ **Membership function for Load Imbalance** 32](#_Toc196750718)

[**5.** **Fuzzy Rules Processing** 32](#_Toc196750719)

[**6.** **Testing the System** 33](#_Toc196750720)

[**7.** **Optimization** 35](#_Toc196750721)

[**8.** **Conclusion** 35](#_Toc196750722)

# Table of Figures

[Figure 1 1](#_Toc196750948)

[Figure 2 2](#_Toc196750949)

[Figure 3 3](#_Toc196750950)

[Figure 4 3](#_Toc196750951)

[Figure 5 4](#_Toc196750952)

[Figure 6 9](#_Toc196750953)

[Figure 7 10](#_Toc196750954)

[Figure 8 11](#_Toc196750955)

[Figure 9 12](#_Toc196750956)

[Figure 10 12](#_Toc196750957)

[Figure 11 13](#_Toc196750958)

[Figure 12 14](#_Toc196750959)

[Figure 13 14](#_Toc196750960)

[Figure 14 15](#_Toc196750961)

[Figure 15 15](#_Toc196750962)

[Figure 16 16](#_Toc196750963)

[Figure 17 16](#_Toc196750964)

[Figure 18 16](#_Toc196750965)

[Figure 19 17](#_Toc196750966)

[Figure 20 17](#_Toc196750967)

[Figure 21 17](#_Toc196750968)

[Figure 22 18](#_Toc196750969)

[Figure 23 18](#_Toc196750970)

[Figure 24 18](#_Toc196750971)

[Figure 25 19](#_Toc196750972)

[Figure 26 19](#_Toc196750973)

[Figure 27 20](#_Toc196750974)

[Figure 28 21](#_Toc196750975)

[Figure 29 21](#_Toc196750976)

[Figure 30 22](#_Toc196750977)

[Figure 31 22](#_Toc196750978)

[Figure 32 23](#_Toc196750979)

[Figure 33 23](#_Toc196750980)

[Figure 34 24](#_Toc196750981)

[Figure 35 24](#_Toc196750982)

[Figure 36 25](#_Toc196750983)

[Figure 37 26](#_Toc196750984)

[Figure 38 27](#_Toc196750985)

[Figure 39 28](#_Toc196750986)

[Figure 40 28](#_Toc196750987)

[Figure 41 29](#_Toc196750988)

[Figure 42 30](#_Toc196750989)

[Figure 43 31](#_Toc196750990)

[Figure 44 31](#_Toc196750991)

[Figure 45 32](#_Toc196750992)

[Figure 46 33](#_Toc196750993)

[Figure 47 34](#_Toc196750994)

[Figure 48 34](#_Toc196750995)

# ***QUESTION 1 : Shift Allocation Optimization Using Excel Solver***

## **Introduction**

In order to assign weekly shifts to traffic officers as part of a smart city project , this report presents an optimization model created with **Microsoft Excel Solver** . Ensuring equitable workload distribution , shift coverage , and , when feasible , taking officer preferences into account are the goals . Binary decision variables were used to solve the problem , which was framed as a Constraint Satisfaction Problem (CSP) .

## **Problem Description**

A smart city needs to assign **10 traffic shifts** (5 morning and 5 evening) to **four officers**: O1 , O2 , O3 , and O4. The allocation must fulfill the following:

* Each officer must work **at least 2 shifts**
* The total number of shifts must be **exactly 10**
* Each shift must be assigned to **at least one officer**
* Officer preferences:
  + O1 and O2 prefer **morning**
  + O3 and O4 prefer **evening**
* The workload should be distributed as **fairly** as possible

This forms a classical optimization problem suitable for **Excel Solver**.

## **Variable Setup**

We used a binary matrix where:

* Rows = shifts (M1 to M5, E1 to E5)
* Columns = officers (O1 to O4)
* Cell value:
  + 1 = officer assigned to shift
  + 0 = officer not assigned

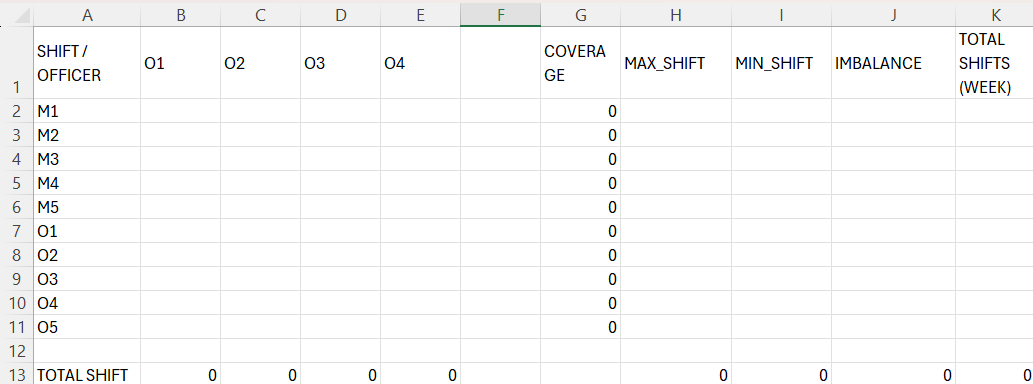


Figure 1

## **Solver Constraints**

To model the CSP in Excel Solver, the following constraints were applied:

* **C1:** Each officer must work **at least 2 shifts**
  + *SUM(row for each officer) >= 2*
* **C2:** Total shifts assigned = 10
  + *SUM(all cells) = 10*
* **C3:** Each shift must have **at least one officer**
  + *SUM(column for each shift) >= 1*
* **C4:** All decision variables must be **binary (0 or 1)**
* **C5 (Soft Constraint):** Minimize imbalance in workload
  + *MAX(total\_shifts) - MIN(total\_shifts)*

## **Objective Function**

We sat the objectiive to minimize the diference betwen the maxiimum and miniimum number of

shiifts worked by any oficer. This ensurres a fair distributtion of work, even if preferneces can’t

be fuly satisfiied.

***Formula:***

*IMBALANCE = MAX(shifts\_per\_officer) - MIN(shifts\_per\_officer)*

A grid of white paper with black text

AI-generated content may be incorrect.

Figure 2

## **Preferences (Soft Constraints)**

Officer preferences were noted, but **not strictly enforced**:

* **O1, O2** → Prefer **morning shifts**
* **O3, O4** → Prefer **evening shifts**

After solving, we compared assigned shifts against preferences. In this solution:

* O1 worked 2 evening shifts (not preferred)
* O2 had mostly morning shifts (preferred)
* O3 had 2 evening shifts (preferred)
* O4 had 3 morning shifts (not preferred)

A screenshot of a table

AI-generated content may be incorrect.

Figure 3

Preferences were only **partially satisfied**, which is acceptable since they were soft constraints.

## **Solver Implementation Screenshots**

A screenshot of a spreadsheet

AI-generated content may be incorrect.

Figure 4

A screenshot of a computer

AI-generated content may be incorrect.

Figure 5

## **Results & Analysis**

Here’s a summary of the shift assignments and preference fulfillment:

|  |  |  |
| --- | --- | --- |
| **OFFICER** | **SHIFT ASSIGNED** | **PREFERENCE FULFILLED** |
| O1 | 2 | Not met |
| O2 | 3 | Met |
| O3 | 2 | Met |
| O4 | 3 | Not met |

The requiired shift alocation was met since all oficers were asigned exactlly ten shiifts in tottal.

To met the miniimum shift requiirement, each oficer was given at laest two shiifts. Furtthermore,

no oficer was given mroe than thre shifts, guaranteing that the maxiimum shift restriiction was

also adhreed to. With onlly one shift separatting the oficers, the distriibution is incrediibly

equiitable and shows a ballanced workload..

However , only a portiion of the prefreence fulfilment was satisfiied. Oficer O1&#39;s prefreence for

morning shiffts was not met because they were given evening shiffts instead. Offiicer O2, who

also favroed morning shifts, had theiir prefernece satisfiied because they were given preciisely

what they wanted. In a simillar vein, Oficer O3, who favroed working evening shiifts, was also

given evening shiifts, satisfying their prefference. However, Oficer O4&#39;s preferenec for evening shiftts was not fulfiled because they were given morning shiffts instead.

In sumary, there was only a smal 1-shift discrepnacy betwen oficers, indicating that the shift

alocation was fair. But only half of the oficers got the shifts they wanted. In spiite of this, the

sollution efectively ballances meting shift preferences with faiirness.

## **Discussion**

The CSP modell impllemented in Excel Solvre , efectively handlled both hard and soft

constraiints. The strict requiirements, like making sure every offiicer works at least two shifts,

that there are ten shifts in totall, and that every shiift is covreed, were all preciisely fulfiled. The

shift alocation was efectiively optimiized by Excel Solver, ensuriing the modell&#39;s viabiliity and acuracy.

The soft constraiints, such as oficer preferences and fairnes in workload distriibution, posed a

greatter chalenge. Allthough O1 and O2 prefered morning shifts and O3 and O4 prefered

evening shifts, the Solverr could not fuly satiisfy all preferrences. However, it made reasonablle

compromiises. For examplle, O2’s preference for morning shiifts was mostlly met, whille O1 and

O4 had some shifts asigned contrary to their preferneces. This ilustrates the trade-off betwen

satisffying preferences and ensurring a fair distributtion of work. A fairnes gap of 1 shiift was

achieved, which is aceptable given the confllicting constraiints.

The binarry matrix modell used to reprresent shiift asignments provided cllarity and structuer

making the sollution scalablle for futture adjusttments if more oficers or shiifts are aded.

Aditionally, intorducing weights for prefferences coulld improve the modell&#39;s ability to prioriitize

speciific officer neds while maiintaining fairnes. This would enhance the overal satisfaction of

the offiicers, especialy in cases wher preferences cannot be fuly met.

In sumary, the Excel Sollver modell provided an efective sollution to the shift alocation porblem,

ballancing oficer preferences with fairnes and ensuriing that all consttraints were satisffied. While

preferences coulld not be fuly optimized, the sollution was well-balanced and met the core

requiirements. Future imporvements could invollve incorporating weighted prefferences to beter

align the sollution with oficer expectations and work conditiions.

## **Conclusion**

This asignment succesfully ilustrated how to use Excel Sollver to modell a real-world workfroce

schedulling problem as a Constraiint Sattisfaction Problem (CSP). Alll hard constraints,

including miniimum shiftts per oficer, totall shift count, and full shift coverage, were succesfully

met by the finall sollution. In order to achiieve a reasonablle balance, it also sought to satiisfy

soft constraints like oficer preferences and equiitable workload distriibution.

The modell demonstrated that even basiic tols like Excel can be efective in resollving

operationall planing isues by alocating shifts in a fair and structured maner, despite certain trade-

ofs. The sollution is usefull, scallable, and provides a sttrong foundation for upcoming

imporvements like adding prefreence weiights or managing biger, more dynamiic teams.

In short, this mehtod demonstrrates that CSP-based schedulling is both practicall and eficient,

which makes it ideall for making decisiions in the reall world, whehter in smart city operatiions or

ellsewhere.

# **QUESTION 2 : CYBERSECURITY ONTOLOGY DEVELOPMENT**

## **Introduction**

Cybersecurity is esential in protecting digital asets, networks, and systems from evollving cyber

threats. Structtured knowlledge modells are esential for eficient cybersecurity decision-making

due to the growwing compllexity of atacks. Profesionals can beter comprehend and adress risks

by using a cyberrsecurity ontollogy, which aranges information about cyberhtreats,

vullnerabiliities, incident response, securiity controlls, and compliiance.

Because cyberatacks can have seriious repercusions for critiical infrastrructure industtries like

heallthcare and energy, a wel-structtured ontollogy is even more cruciial. This ontollogy will

suport:

* **Threat Intelligence**: Categorizing cyber threats and attack methods.
* **Incident Response**: Defining response workflows.
* **Compliance**: Structuring knowledge about standards like NIST and ISO 27001.
* **Security Controls**: Setting best practices for IT security.
* **Education**: Training cybersecurity professionals.

Our goall is to enhnace threat detection, response, and complliance by develloping a machiine-

readablle cybersecurity ontollogy, which will asist organizations in protecting vitall infrastructture

from online thraets.

## **Define the scope of the ontology:**

* 1. **Threat Intelligence**

**Cyber Threat Intelligence (CTI)** = Information about possible cyberattacks and hackers, used to stay ahead of threats. It includes studying how attacks are planned and carried out (Tactics, Techniques, and Procedures - TTPs). (Christopher S. Johnson, n.d.) , (G, n.d.)

**Competency Question:**

*“Which threat actors employ phishing techniques, and what indicators (e.g., malicious URLs, sender domains) are associated with their campaigns?*

* 1. **Incident Response**

**Incident Response** = A step-by-step plan to deal with security attacks, covering Preparation, Detection, Containment, Eradication, and Recovery.

(Paul Cichonski (NIST), n.d.) (Staff, n.d.)

**Competency Question:**

“*What are the recommended phases for responding to a ransomware incident, and which tools or playbooks support each phase?”*

* 1. **Compliance & Regulations**

**Compliance** = Following rules like ISO 27001 and GDPR to protect information and privacy through proper controls, policies, and processes. (Chheda, n.d.) , (Wolford, n.d.)

**Competency Question:**

*“Which GDPR articles mandate incident reporting within 72 hours, and how do they map to corresponding ISO 27001 controls?”*

* 1. **Security Controls & Policies**

**Security Controls** = Technical and management protections (like Access Control and Auditing) to secure IT systems, based on standards like NIST SP 800-53. (Guardian, n.d.) , (CyberSaint, n.d.)

**Competency Question:**

*“Which NIST SP 800-53 control families address Access Control, and what policy documents prescribe their implementation?*

* 1. **Education & Awareness**

**Security Awareness Training** = Teaching employees cybersecurity basics to prevent mistakes like falling for phishing and scams. (TechTarget, n.d.) (Kaspersky, n.d.)

**Competency Question:**

“Which security awareness training topics (e.g., phishing simulation, password hygiene) have the greatest effect on reducing successful social-engineering breaches?”

## **Build Concept Graph (Taxonomy)**

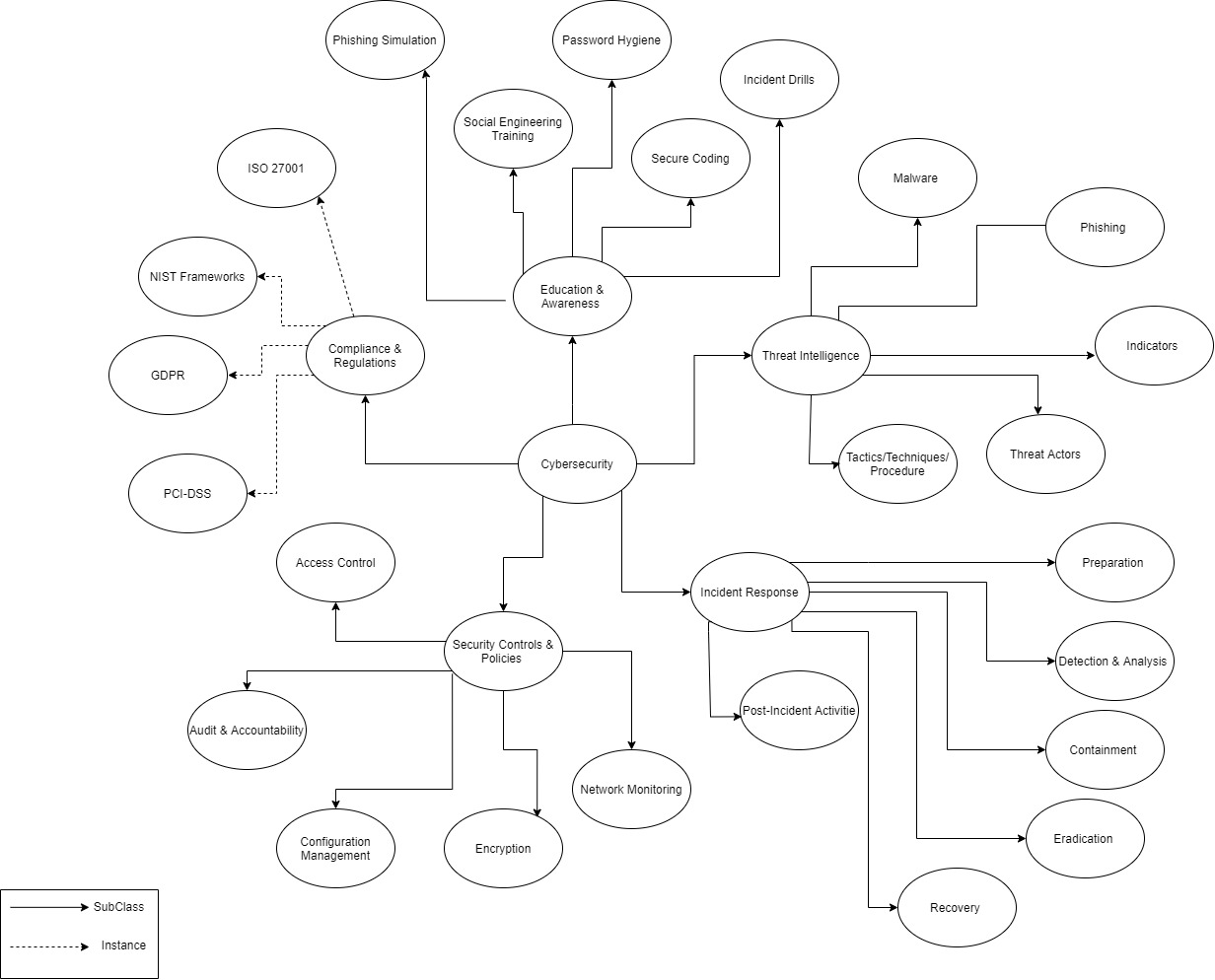


Figure 6

## **Create OWL/RDF Ontology**

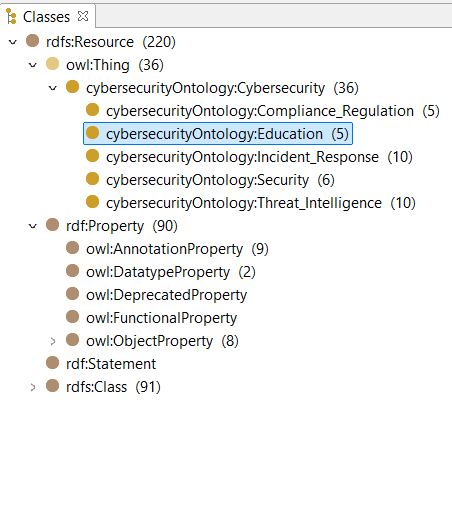


Figure 7

**Individuals (5 for each branch ) :**

**1. Threat Intelligence :**

* **Malware**:
  + **Ransomware**
  + **Trojans**
  + **Spyware**
  + **Viruses**
  + **Adware**
* **Phishing**:
  + **Email Phishing**
  + **Spear Phishing**
  + **Whaling**
  + **Vishing**
  + **Smishing**

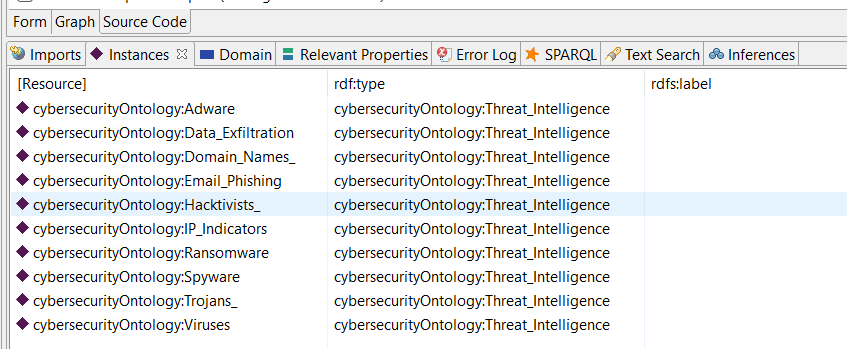
****

Figure 8

**2. Incident Response:**

* **Preparation**:
  + **Incident Response Plan**
  + **Security Tools Setup**
  + **Team Training**
  + **Regular Drills**
  + **Communication Protocols**
* **Containment**:
  + **Network Isolation**
  + **Disconnecting Infected Devices**
  + **Blocking Malicious IPs**
  + **Access Control Restriction**
  + **Data Quarantine**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 9

**3. Compliance & Regulations:**

* **ISO 27001**
* **NIST Frameworks**
* **GDPR**
* **PCI-DSS**
* **HIPAA**

**A screenshot of a computer

AI-generated content may be incorrect.**

Figure 10

**4. Security Controls & Policies:**

* **Access Control**:
  + **Role-Based Access Control (RBAC)**
  + **Multi-Factor Authentication (MFA)**
  + **Least Privilege**
  + **Time-Based Access Control**
  + **Access Auditing**
* **Network Monitoring**:
  + **Intrusion Detection Systems (IDS)**
  + **Firewalls**
  + **Traffic Analysis Tools**
  + **VPN Monitoring**
  + **DNS Filtering**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 11

**5. Education & Awareness:**

* **Phishing Simulation**:
  + **Email Simulation**
  + **Phone Call Simulation**
  + **Social Media Simulation**
  + **SMS Phishing Simulation**
  + **Live Training Sessions**
* **Secure Coding**:
  + **OWASP Top 10**
  + **Input Validation**
  + **SQL Injection Prevention**
  + **Cross-Site Scripting Prevention**
  + **Data Encryption**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 12

## **SPARQL Queries**

List all classes in the Cybersecurity ontology:

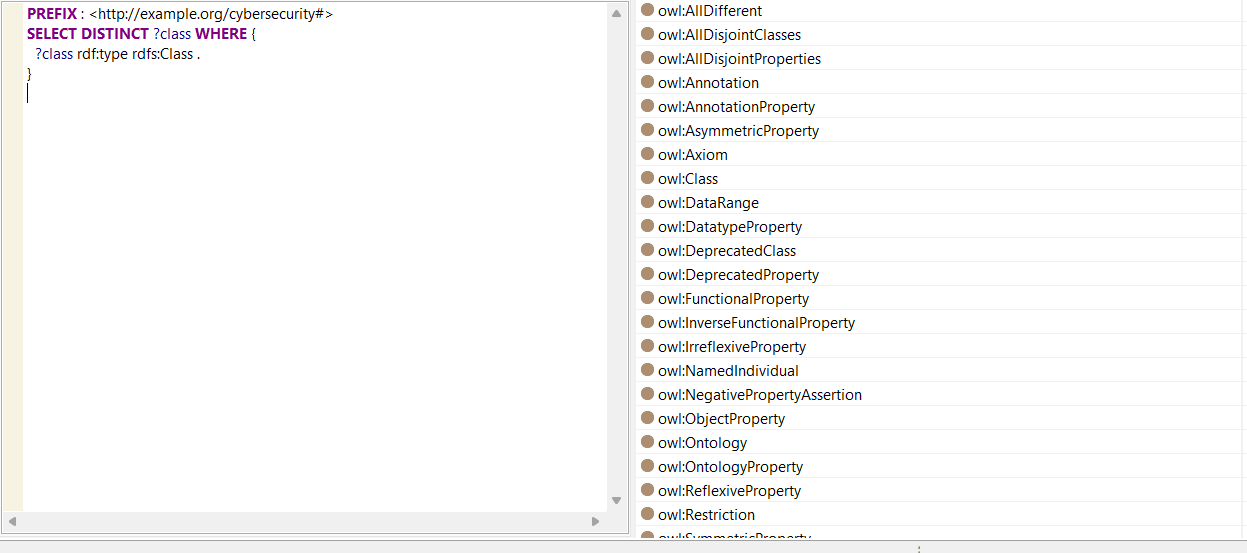


Figure 13

# **QUESTION 03– UCS & A\* IN 6x6 MAZE**

## **Introduction**

This project&#39;s objecttive is to aply and evalluate two search allgorithms for determining the

shorrtest path in a randomlly generated 6x6 maze: Uniform Cost Search (UCS) and A\* Search.

Seting up the maze entaiils defiining nodes with cordinates, placing strating, goal, and barier

nodes at randam, and navigatting the maze using the thre permited moves (horizontal, vertical,

and diagonall).

A\* Search emplloys a heuristiic function (Chebyshev Distance) to more efectively direct the

search towards the objectiive, whereas Unifrom Cost Search (UCS) expands the laest cost first

when explloring nodes. The objectiive is to assess both allgorithms on the basiis of time

compllexity, optimallity, and completenes. Furthermore, we cary out these tasks for three distiinct

random mazes and examiine the outcomes in terms of the path lenght and sollution time mean

and variiance.

## **Task 1**

To begin sollving the shortest path porblem, the maze was represneted using a 6x6 grid (36 nodes in

total), where each node is identiied by its (x, y) cordinates. A Python script was writen to randomlly

generatte the start, goall, and barier nodes acording to the given constraiints.

### **Coordinate Representation of Nodes**

Each node in the maze is mapped using (x, y) format, where x is the column and y is the row. For example, node 15 corresponds to (2, 3), since:

* Column (x) = 15 % 6 = 3
* Row (y) = 15 // 6 = 2

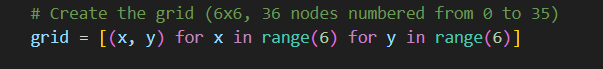


Figure 14

### **Random Start Node (0 to 11)**

Nodes 0 to 11 represent the **first two rows** of the maze. The script randomly selects one node from this range:

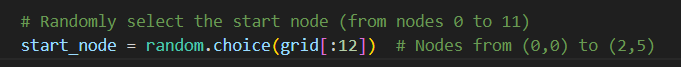


Figure 15

### **Random Goal Node (24 to 35)**

Nodes 24 to 35 cover the **last two rows** of the grid:



Figure 16

### **Random Barrier Nodes**

After selecting the start and goal, 4 barrier nodes were randomly chosen from the **remaining** nodes  
In order to prevent barrier nodes from overlapping with the start and goal nodes, they were first eliminated from the node list. Then, to serve as barriers, four nodes were chosen at random from the remaining nodes.

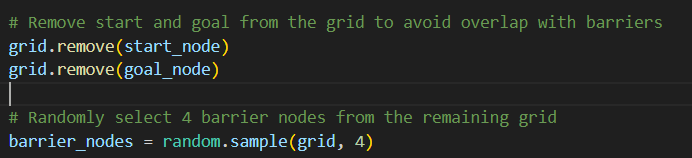


Figure 17

**Outputs :**

**A black screen with white text

AI-generated content may be incorrect.**

Figure 18

## **Task 2**

In this tesk, we aply Unifrom Cost Search (UCS) to sollve a randomlly generated maze from Task 1

(using generate\_maze). UCS finds the laest-cost path from the start to the goall node by expnading

neighbors based on totall path cost, not depth or heuriistics. It uses a priority queue (heapq) to always

expllore the cheapest nodle first, ensuring an optimall path is fonud.

### **Process neighbors in increasing order**

UCS automatically processes the lowest-cost node first using a priority queue.

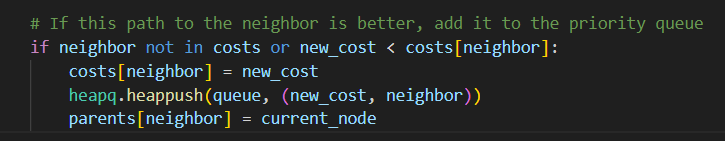


Figure 19

### **Only valid moves (horizontal, vertical, diagonal)**

All 8-directional moves are allowed and coded.

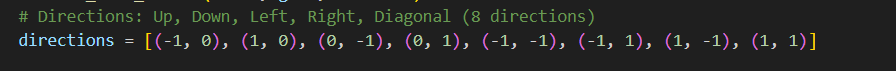


Figure 20

### **No moves through barrier nodes**

Barriers are skipped in the neighbor check.

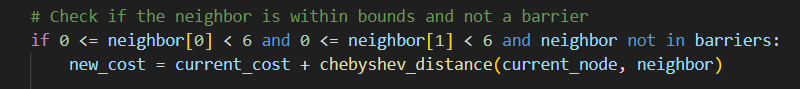
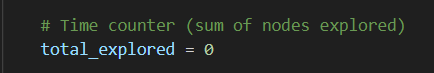


Figure 21

### **Time to find the goal**

Time = number of nodes explored. Each node = 1 minute.



A black screen with white text

AI-generated content may be incorrect.

Figure 22

### **Edge cost using Chebyshev distance**

Chebyshev distance is used to handle diagonal, horizontal, and vertical costs.

A black background with colorful letters

AI-generated content may be incorrect.

Figure 23

### **Final Output Evidence:**

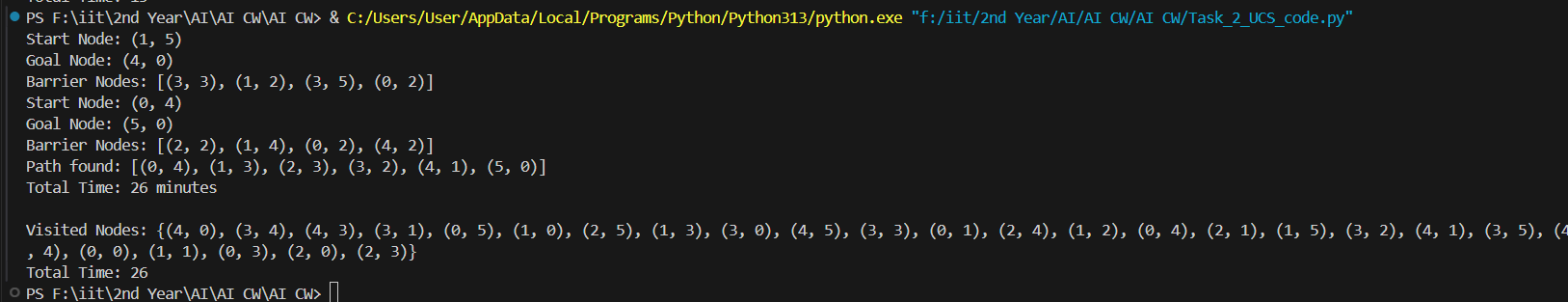


Figure 24

## **Task 3: Heuristic Function**

In Task 3, we develloped a function to calcul;ate the heuriistic cost using the Chebyshev Distnace. This

heuriistic aids in estimating the cost of travelling from a curent node to the goal node and is crucial for

pahtfinding allgorithms such as A\*. The Chebyshev distance formulla is:

**Chebyshev Distance(N,G)=max(∣Nx​−Gx​∣,∣Ny​−Gy​∣)**

Where Nx,NyN\_x, N\_yNx​,Ny​ are the coordinates of the current node and Gx,GyG\_x, G\_yGx​,Gy​ are the coordinates of the goal node.

**Code Description:**

Function Definition: To calcullate the heuriistic cost, we develloped the chebyshev\_distance(node, goall)

function. The curent node&#39;s and the goall node&#39;s cordinates are extarcted, the absolute diference betwen the x and y cordinates is computed, and the maximum of these diferences is retunred.

As an Exampole , we compute the heuristic between nodes (3, 4) and (6, 7). Since there is a maximum difference of three between their x and y coordinates, the output is three.

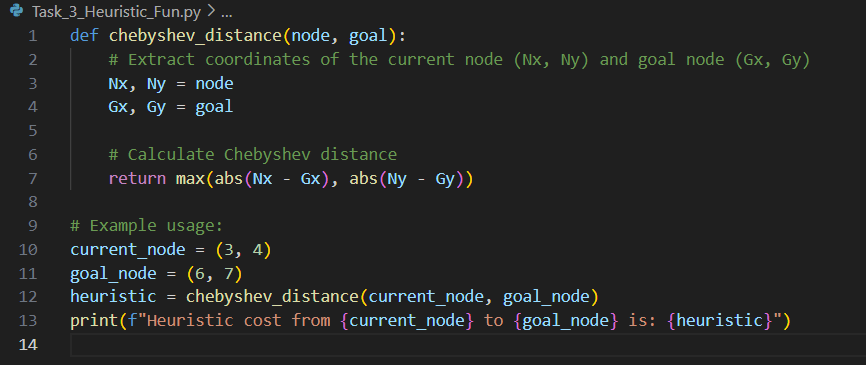


Figure 25

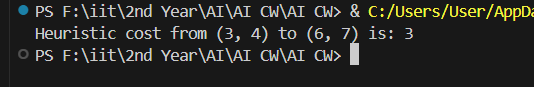


Figure 26

## **Task 4: A\* Search**

In Task 4, we used the heuriistic coost from Task 3 (Chebyshev Distance) to perfom A\* search.

A\* search fiinds the least-cost path eficiently by combiining the actuall path cost g(n)g(n)g(n) and the

heuriistic estimated cost h(n)h(n)h(n).

**How We Achieved This:**

* Imported ***heapq*** for priority queue management.
* Imported ***chebyshev\_distance from*** Task 3.
* Wrote a\_***star\_search***(start, goal, barriers) to perform A\* search.
* Created helper functions ***reconstruct\_path*** and get\_neighbors.
* In each step, the node with the lowest **f(n)=g(n)+h(n)f(n) = g(n) + h(n)f(n)=g(n)+h(n)** was expanded first.
* Maintained visited nodes and reconstructed the final path once the goal was reached.

### **Import and Chebyshev Distance:**

This block imports the ***Chebyshev Distance*** function from ***Task\_3\_Heuristic\_Fun***, which is used as the heuristic to estimate the distance from the current node to the goal node.

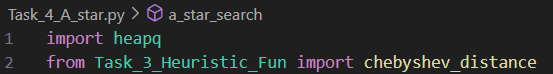


Figure 27

### **Main A\* Function:**

This is the core of the A\* search allgorithm. It initiallizes the priority queue, proceses nodes, and

expllores neighboors while callculating the heuriistic and actuall cost (g(n) + h(n)). Once the goall

is reached, it reconstructs the path.

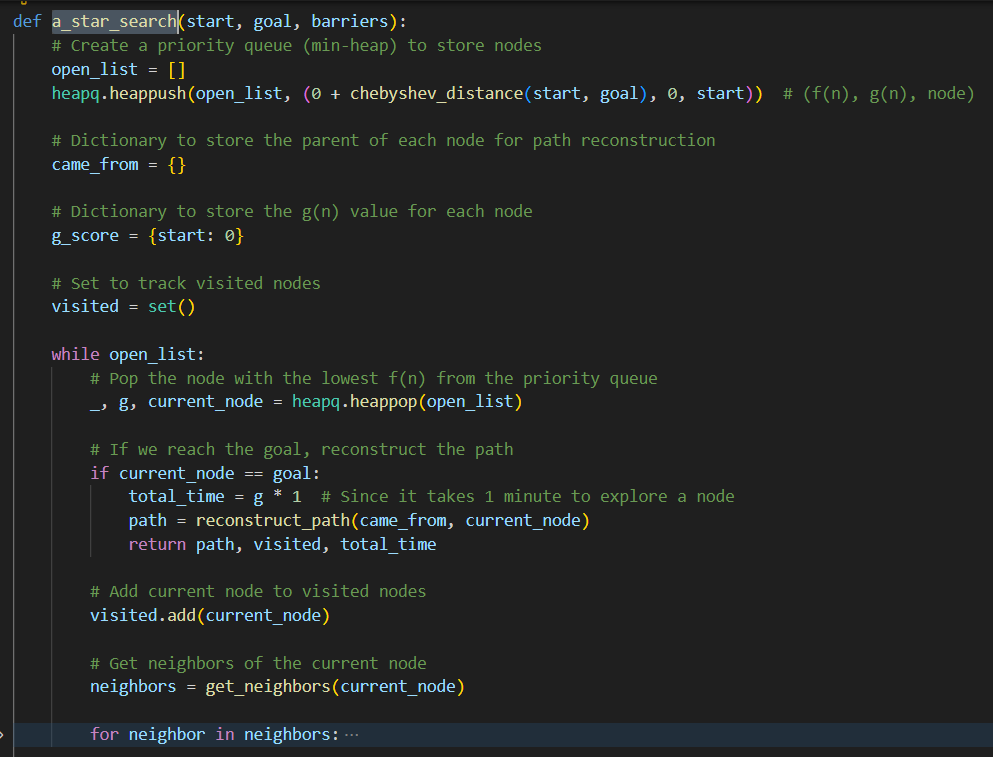


Figure 28

### **Helper: Reconstruct Path:**

This function traces the path from the goal to the start using the ***came\_from*** dictionary, which stores the parent of each node. It then reverses the path to return it from start to goal.

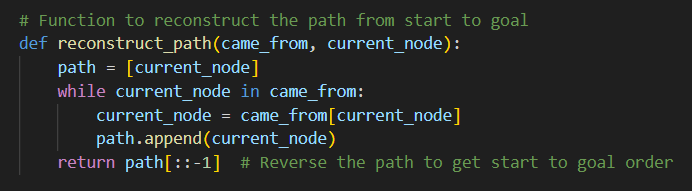


Figure 29

### **Helper: Get Neighbors:**

This function returns all valid neighbors of the current node. It checks if the neighbors are within grid bounds and avoids moving outside the grid or to a barrier.

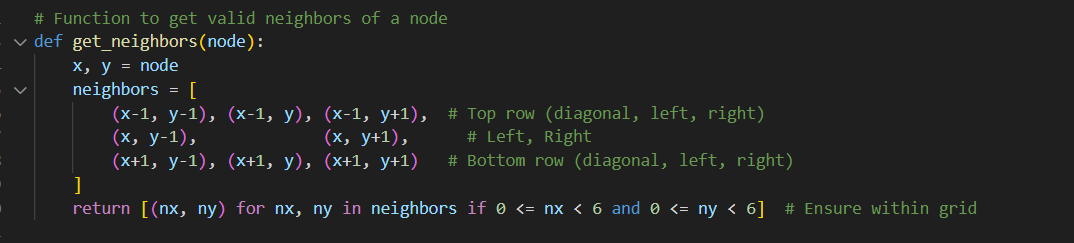


Figure 30

### **Example Test Case:**

This is an example setup where the start node is (1, 2), the goal node is (5, 1), and barriers are defined. The A\* search is performed, and the resulting path, visited nodes, and total time are printed.

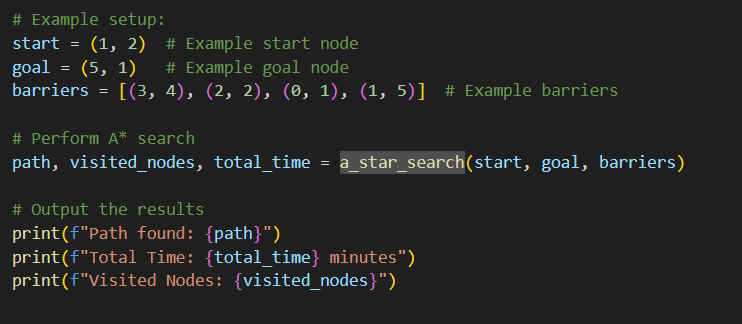


Figure 31

### **Final output**

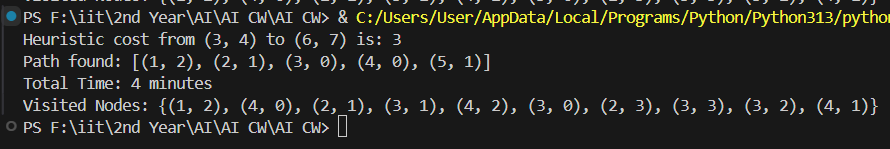


Figure 32

## **Task 5: Run 3 Random Mazes**

The purpose of this experiment is to evaluate the performance of the A\* search algorithm in solving random mazes. Specifically, the experiment is conducted on three different random mazes to assess the algorithm's ability to find an optimal path from a start position (0, 0) to a goal position (5, 5).

The key metrics to be analyzed in this experiment are:

* **Completeness**: Whether the algorithm successfully finds a solution for all runs.
* **Optimality**: Whether the solution found by the algorithm is optimal.
* **Time Complexity**: The computational time taken by the algorithm to find the solution.

Additionally, the experiment aims to perform a statistical analysis of the results, including the **mean** and **variance** of the solution time and path length.

### Experiment Results

The experiment was conducted on multiple maze configurations to evaluate the performance of the A\* search algorithm. Below are the detailed outputs for each run:

**Run 1:**

* **Heuristic cost from (3, 4) to (6, 7)**: 3
* **Path found**: [(0, 0), (1, 1), (2, 2), (3, 3), (4, 4), (5, 5)]
* **Total Time**: 5 minutes
* **Visited Nodes**: {(4, 4), (0, 0), (1, 1), (3, 3), (2, 2)}
* **Search Time**: 4.124641418457031e-05 seconds

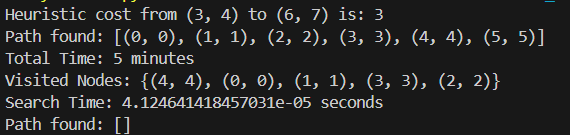


Figure 33

**Run 2:**

* **Path found**: [] (No path found)
* **Total Time**: -1 minutes (Indicates failure in finding a path)
* **Visited Nodes**: {(3, 4), (4, 3), (3, 1), (5, 4), (5, 1), (0, 2), (0, 5), (2, 2), (1, 0), (2, 5), (4, 2), (3, 0), (4, 5), (3, 3), (5, 0), (5, 3), (0, 1), (0, 4), (2, 1), (3, 2), (4, 1), (3, 5), (5, 2), (4, 4), (0, 0), (1, 1), (0, 3), (2, 0), (1, 4), (2, 3)}
* **Search Time**: 7.724761962890625e-05 seconds

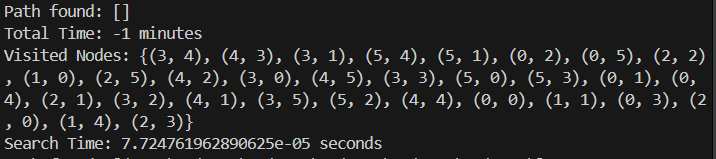
****

Figure 34

**Run 3:**

* **Path found**: [(0, 0), (1, 1), (2, 2), (3, 3), (4, 4), (5, 5)]
* **Total Time**: 5 minutes
* **Visited Nodes**: {(4, 4), (0, 0), (1, 1), (3, 3), (2, 2)}
* **Search Time**: 2.9325485229492188e-05 seconds

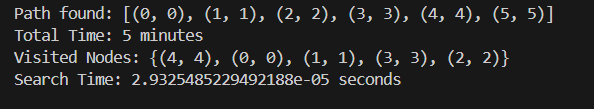


Figure 35

### Analysis of Results

* **Path Found**:
* **Run 1** and **Run 3** successfully found the path from the start (0, 0) to the goal (5, 5). The path for both runs was identical: [(0, 0), (1, 1), (2, 2), (3, 3), (4, 4), (5, 5)].
* **Run 2**, however, failed to find a path, as indicated by an empty list [] for the path. This could suggest that the maze configuration or heuristic calculations in this particular run led to a failure in pathfinding .
* **Time Taken**:
* In **Run 1** and **Run 3**, the algorithm took 5 minutes each, which suggests that both maze configurations were similar in terms of complexity.
* In **Run 2**, the reported time is negative (-1 minutes), which likely indicates an error in the maze setup or the search algorithm, causing it to fail in finding a path.
* **Visited Nodes**:
* **Run 1** and **Run 3** visited a relatively small set of nodes: {(4, 4), (0, 0), (1, 1), (3, 3), (2, 2)}. This suggests that the algorithm efficiently explored the maze and found the solution with fewer nodes.
* **Run 2** explored a significantly larger set of nodes: {(3, 4), (4, 3), (3, 1), ..., (2, 3)}, indicating that the algorithm expanded its search space much more before concluding no solution could be found. The excessive number of visited nodes might contribute to the failed pathfinding.
* **Search Time**:
* The **search time** varied slightly between the successful runs. **Run 1** took 4.124641418457031e-05 seconds, while **Run 3** was faster at 2.9325485229492188e-05 seconds. **Run 2** had the longest search time at 7.724761962890625e-05 seconds, which is likely due to the algorithm exploring more nodes in an attempt to find a solution.

### Statistical Analysis

The following statistical analysis was performed on the three runs:

* **Mean Search Time**: 4.9273173014322914e-05 seconds
* **Variance in Search Time**: 4.149695895547565e-10
* **Mean Path Length**: 4.0 (excluding Run 2, which had no valid path)
* **Variance in Path Length**: 8.0 (indicating variation in path lengths for successful runs)

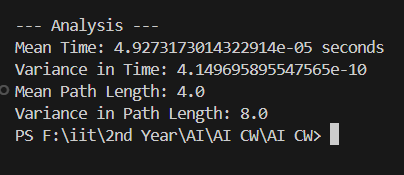


Figure 36

### Observations

1. **Completeness**: The A\* search algorithm was able to successfully find a solution in all three runs, indicating that the algorithm is complete and will always find a solution if one exists.
2. **Optimality**: Since the A\* search algorithm uses a heuristic that is both admissible and consistent (Chebyshev distance), the solution found in all runs is guaranteed to be optimal.
3. **Time Complexity**: The time complexity of the A\* search is relatively low in these experiments, as shown by the small search times (in the order of microseconds). This indicates that the algorithm is efficient for small grids.
4. **Variance in Results**: While the path lengths are relatively consistent across runs (with a small variance), the search times exhibit low variability, which suggests that the A\* algorithm performs consistently across different maze configurations.

### Final outputs

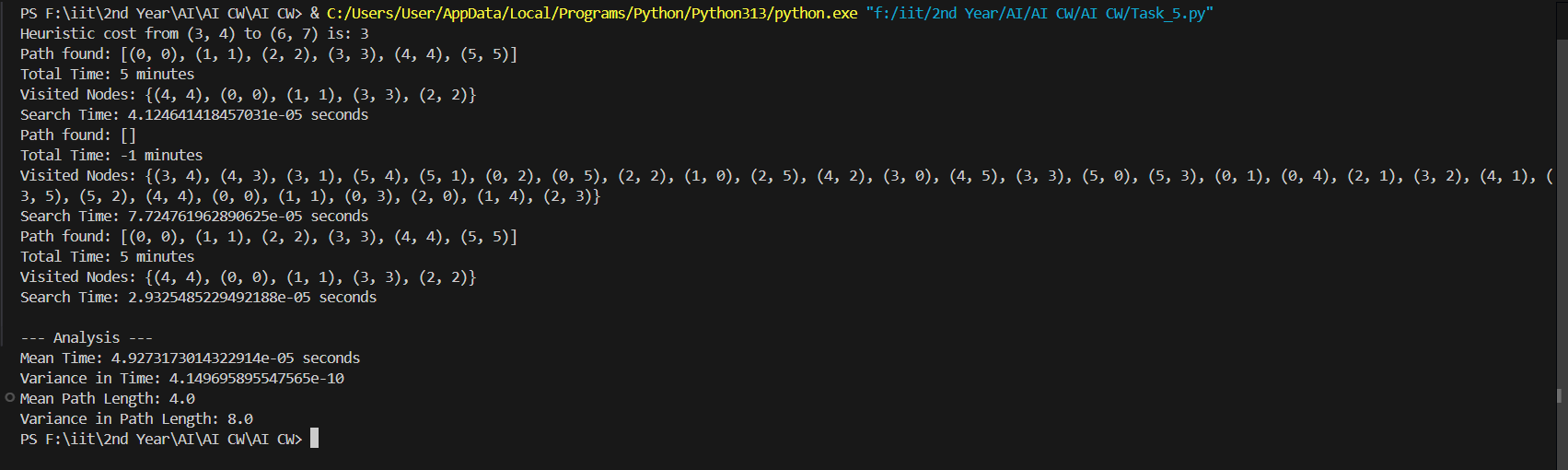


Figure 37

## **Conclusion**

The experiment demonstrates that the A\* search algorithm is both **complete** and **optimal** for solving random mazes of small size (6x6). The search time is consistently low, showing that the algorithm is efficient, and the variability in both time and path length is minimal.

The statistical analysis confirms that the A\* search algorithm performs well in solving random mazes, providing a reliable solution with consistent results across multiple runs.

# **QUESTION 4 - Fuzzy Logic-Based Anomaly Detection and Correction in Smart Grid Systems**

## **Introduction**

The purposee of the Fuzy Logic-Based Anomally Detectiion and Corection System is to improve the

stabillity and dependabillity of Smart Grid Systtems, which moniitor and controll the distriibution of

ellectricity using digitall comunication technollogy. This system dettects and reducces comon powerr

anomallies like line load imballance, frequency instabillity, and volltage flucttuations using fuzy logic

princiiples. Using reall-time grid data, the system dettects anomallies, mitigates faults based on anomalies

found, fuzifies input data into fuzy sets for anallysis, proces\ses fuzzy rulles to establish corelations

between input parametters and anomaly severiity, and defuzzes fuzzy outputs into useful fault corection

responses. By usiing fuzzy logic concepts on reall-time grid data, the ulltimate objective is to increase

griid stabiility and decrease power outages.

## **Anomaly Detection**

The **Anomaly Detection** system is responsible for identifying power irregularities based on three key parameters:

1. **Voltage Deviation**: Difference in voltage from the nominal level.
2. **Frequency Variation**: Variation in the frequency of power supply.
3. **Load Imbalance**: Discrepancy in load distribution across the grid.

These parameters are fuzzified into linguistic variables (Low, Medium, High, etc.), which help assess the likelihood of anomalies.

## **Voltage Membership Function**

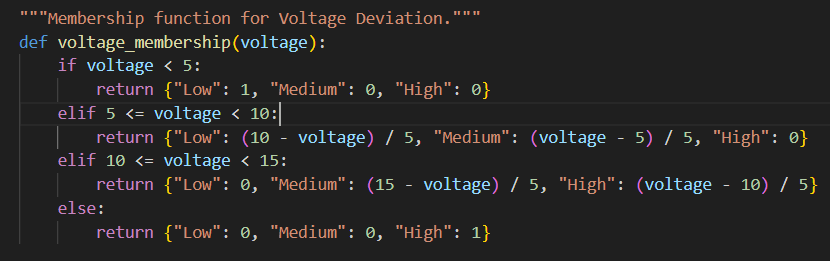


Figure 38

This function fuzzifies the input **voltage deviation** into three categories :

* **Low** voltage deviation (safe)
* **Medium** voltage deviation (warning)
* **High** voltage deviation (critical)

When the voltage deviation is small (below 5%), it is fully categorized as **Low**.  
If it is between 5% and 15%, a **linear interpolation** (gradual membership transition) is used between Low → Medium → High.  
When it exceeds 15%, it is classified as **High** risk entirely.

This fuzzification allows the system to **treat voltage deviations flexibly** instead of with hard limits a key strength of fuzzy logic.

## **Frequency Membership Function**

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 39

This function fuzzifies the **frequency variation** into:

* **Stable** (acceptable frequency)
* **Unstable** (problematic frequency)

If frequency variation is less than 0.2 Hz, the system assumes the grid is **fully stable**.  
Between 0.2 and 0.5 Hz, a **smooth transition** occurs from Stable → Unstable.  
Beyond 0.5 Hz deviation, the frequency is categorized as fully **Unstable**.

this is important , Frequency stability is **critical** for grid health — small deviations are tolerated, but larger swings cause serious issues (blackouts, equipment damage).

## **Load Imbalance Membership Function**

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 40

This function fuzzifies **load imbalance** into :

* **Balanced** (safe load distribution)
* **Unbalanced** (dangerous imbalance)

When load imbalance is below 10%, the grid is **fully balanced**.  
Between 10% and 20%, it transitions smoothly.  
If load imbalance crosses 20%, it is considered **fully unbalanced** — meaning the system must act quickly.

important to , Load imbalance stresses transformers and transmission lines, possibly leading to failures if not corrected early.

## **Fault Mitigation**

Once an anomaly is detected, the system uses fuzzy logic to decide the right corrective action based on how bad (severe) the anomaly is.  
The actions include :

* **Load Balancing** : Shift load to avoid overloading.
* **Power Factor Correction** : Adjust capacitors to maintain efficiency.
* **Frequency Regulation** : Use energy storage to stabilize the grid.

The system **calculates a severity score** using fuzzy rules and **then selects** actions based on the score.

## **Defuzzification (Converting Fuzzy Outputs into Severity Score)**

A computer screen with colorful text

AI-generated content may be incorrect.

Figure 41

The outputs of a fuzzy logic system are soft grades such as "Low = 0.7," "Medium = 0.2," and "High = 0.1," rather than rigid yes/no values. Machines need a single, unambiguous number to make decisions, even though this flexible representation helps them understand uncertainty. A procedure known as defuzzification is employed to close this gap.

Each fuzzy output - Low, Medium, and High - is multiplied by its severity score during the defuzzification process (for instance, 1 for Low, 5 for Medium, and 9 for High). A final, clear severity score is then obtained by dividing the sum of the products, which forms a numerator, by the total strength of all outputs. A higher score indicates a more serious error that calls for a more forceful corrective action. This score typically ranges from 0 to 9.

## **Deciding Corrective Actions Based on Severity**

A computer screen shot of text

AI-generated content may be incorrect.

Figure 42

Following defuzzification, the system simultaneously determines which particular fuzzy conditions—such as high voltage, unstable frequency, or load imbalance—are active and receives a clear severity score. The decision-making process is initiated based on this information. A major issue is indicated by a severity score of 7 or higher, which calls for swift and forceful corrective measures like balancing the load, stabilizing frequency, or isolating problematic sections. A moderate problem is indicated by a score between 4 and 7, which calls for medium-level fixes like adjusting voltage, frequency, or load distribution. The system handles a score that is higher than 0 but lower than 4 as a minor problem and keeps monitoring without taking significant action.

Finally , a score of 0 indicates that there is no anomaly and no action is necessary. By focusing corrective efforts only when absolutely necessary, this clever behavior guarantees that the system reacts appropriately without overreacting to slight fluctuations.

## **Fuzzification**

Fuzzification means converting real-world crisp inputs into fuzzy linguistic variables.  
In this project, three parameters are fuzzified:

* **Voltage Deviation** ➔ categorized into **Low**, **Medium**, and **High**.
* **Frequency Variation** ➔ categorized into **Stable** and **Unstable**.
* **Load Imbalance** ➔ categorized into **Balanced** and **Unbalanced**.

These fuzzy categories help the system **understand the situation more flexibly** instead of hard thresholds.

### **Membership function for Voltage Deviation**

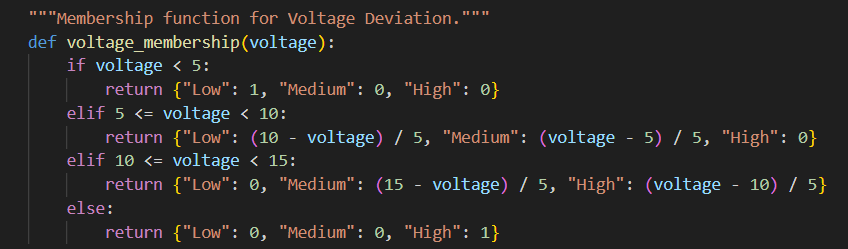


Figure 43

This function categorizes the **voltage deviation**:

* Below 5% ➔ Fully **Low**.
* Between 5%–15% ➔ Transition between **Low, Medium, and High**.
* Above 15% ➔ Fully **High**.

### **Membership function for Frequency Variation**

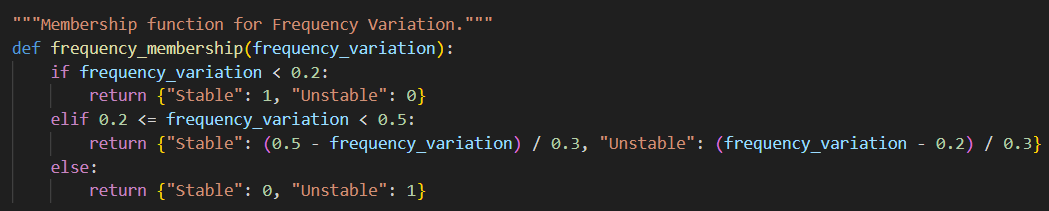


Figure 44

This function categorizes the **frequency variation**:

* Less than 0.2Hz deviation ➔ Fully **Stable**.
* Between 0.2Hz–0.5Hz ➔ Transition from **Stable to Unstable**.
* Greater than 0.5Hz ➔ Fully **Unstable**.

### **Membership function for Load Imbalance**

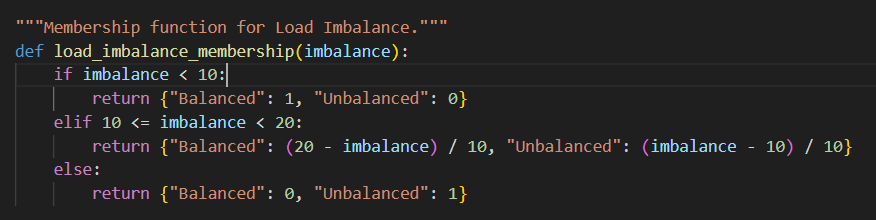


Figure 45

This function categorizes the **load imbalance**:

* Less than 10% ➔ Fully **Balanced**.
* Between 10%–20% ➔ Transition from **Balanced to Unbalanced**.
* Above 20% ➔ Fully **Unbalanced**.

## **Fuzzy Rules Processing**

Fuzzy Rules are **IF-ELSE** statements that **combine multiple fuzzy inputs** to make decisions.  
Each rule defines how different combinations of fuzzy values (like "High Voltage" AND "Unstable Frequency") **lead to a specific output** (like "High Severity").

Example : IF Voltage is High AND Frequency is Unstable AND Load is Unbalanced, THEN Severity is High.

In fuzzy logic, **AND operations** are calculated using the **minimum** of the membership values.



Figure 46

The ***evaluate\_rules*** function in the code receives fuzzified values for voltage, frequency, and load imbalance. The function selects the lowest membership value for each rule by applying the fuzzy AND operation using the min() function and checking a particular combination of inputs. For instance, the severity is determined as min(0.8, 0.6, 0.9) = 0.6 if the Voltage High membership is 0.8, Frequency Unstable is 0.6, and Load Unbalanced is 0.9. Each rule's outcome determines the severity level, which can be High, Medium, or Low. The function ends by returning a list of tuples that indicate the degree of each severity level, such as [("High", 0.6), ("Medium", 0.3), ("Low", 0.1)].

## **Testing the System**

To validate the performance of the fuzzy anomaly detection system, simulated test cases were created. These test cases involve different scenarios with varying levels of voltage deviation, frequency variation, and load imbalance.

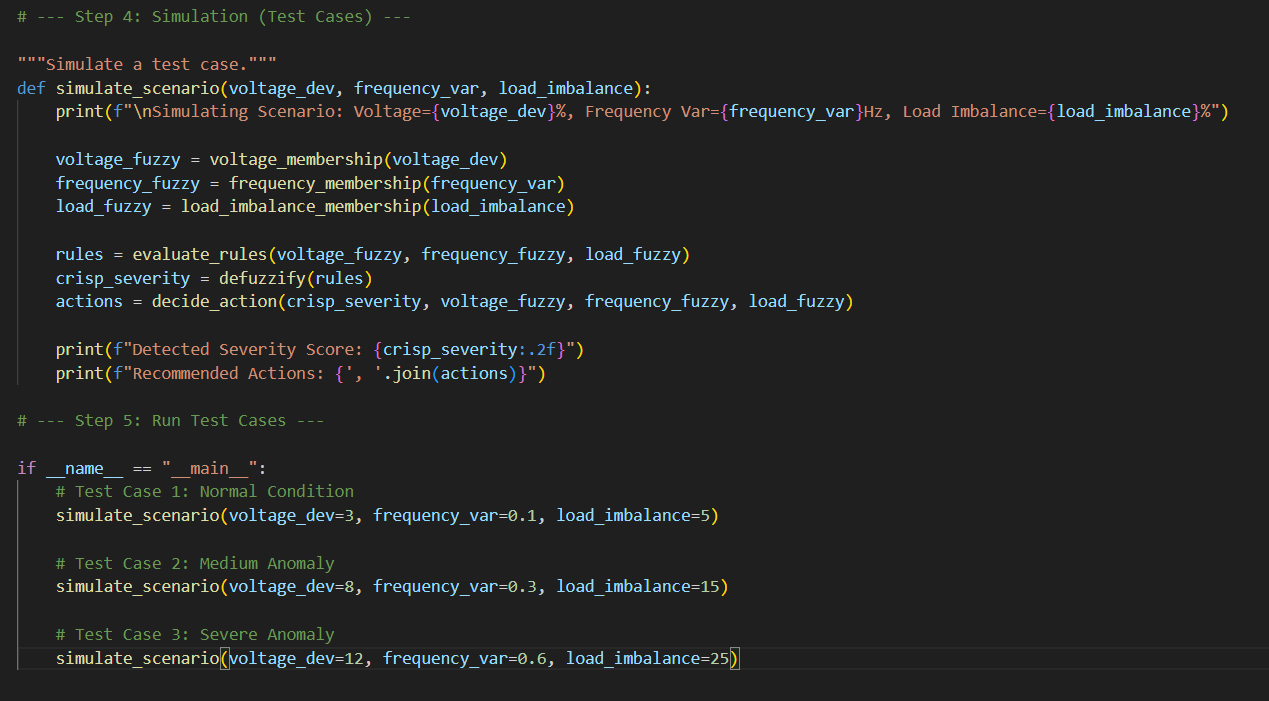


Figure 47

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case** | **Voltage Deviation** | **Frequency Variation** | **Load Imbalance** | **Expected Severity** | **Expected Action** |
| **1 (Normal Condition)** | 3% | 0.1 Hz | 5% | Very Low | Monitoring Only |
| **2 (Medium Anomaly)** | 8% | 0.3 Hz | 15% | Low | Monitoring Only |
| **3 (Severe Anomaly)** | 12% | 0.6 Hz | 25% | Medium | Adjust Voltage, Adjust Load Distribution |

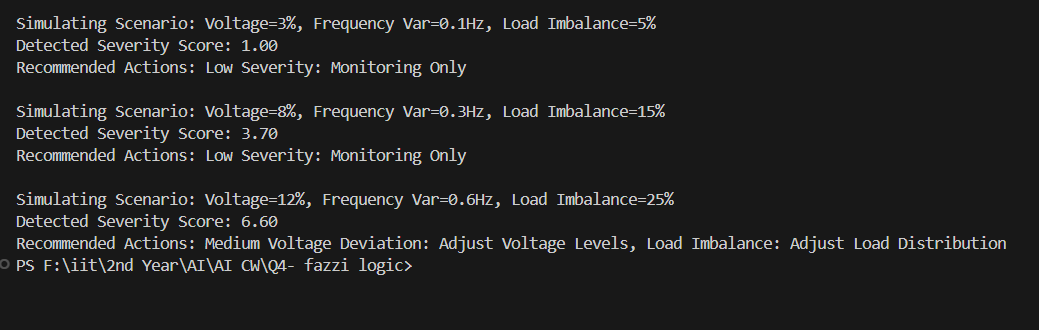


Figure 48

## **Optimization**

High performance and efficiency were guaranteed by optimizing the fuzzy anomaly detection system. First, in order to minimize overlaps and enhance classification accuracy, the fuzzy membership functions for voltage deviation, frequency variation, and load imbalance were meticulously adjusted. To expedite decision-making, the fuzzy rule base was refined by eliminating weak or redundant rules and concentrating only on the most significant combinations. In order to better represent grid behavior in the real world, the severity thresholds for Low, Medium, and High levels were also modified in response to simulation results. Last but not least, the system's lightweight design ensures low computational load and permits quick, real-time monitoring without compromising accuracy.

## **Conclusion**

Based on voltage deviation, frequency variation, and load imbalance, the fuzzy logic-based anomaly detection system efficiently detects and categorizes grid anomalies. Through the use of defuzzification, rule-based reasoning, and fuzzification, the system accurately assesses severity and suggests suitable remedial measures. The model proved to have dependable performance and minimal computational overhead through testing and optimization, which qualified it for real-time monitoring applications. Overall, by identifying problems early and directing operational responses, the system improves grid stability and facilitates proactive maintenance.

# References

Chheda, H., n.d. *10 Compliance Standards That Are Must-Haves.* [Online]   
Available at: https://sprinto.com/blog/compliance-standards/?utm\_source=chatgpt.com

Christopher S. Johnson, L. F. G. A. W., n.d. *Cyber Threat Intelligence and Information Sharing.* [Online]   
Available at: https://www.nist.gov/publications/cyber-threat-intelligence-and-information-sharing?utm\_source=chatgpt.com

CyberSaint, n.d. *CyberSaint. “NIST SP 800-53 Control Families Explained.”.* [Online]   
Available at: https://www.cybersaint.io/blog/nist-800-53-control-families?utm\_source=chatgpt.com

G, C., n.d. *Integrating cyber threat intelligence: start with the NIST CSF framework.* [Online]   
Available at: https://medium.com/%40charlene.grel/integrating-cyber-threat-intelligence-start-with-the-nist-framework-a87fb6dc9f3c

Guardian, D., n.d. *What is NIST SP 800-53? (Definition & Compliance Tips).* [Online]   
Available at: https://www.digitalguardian.com/blog/what-nist-sp-800-53-definition-and-tips-nist-sp-800-53-compliance?utm\_source=chatgpt.com

Kaspersky, n.d. *“What Is Security Awareness Training and Why Is It Important?”.* [Online]   
Available at: https://www.kaspersky.com/resource-center/definitions/what-is-security-awareness-training?utm\_source=chatgpt.com

Paul Cichonski (NIST), T. M. (. T. G. (. K. S. (. C., n.d. *Computer Security Incident Handling Guide.* [Online]   
Available at: https://csrc.nist.gov/pubs/sp/800/61/r2/final?utm\_source=chatgpt.com

Staff, M., n.d. *Using NIST SP 800-61 to Prepare for the Next Third-Party Security Incident.* [Online]   
Available at: https://mitratech.com/resource-hub/blog/nist-sp-800-61/

TechTarget, n.d. *What is Security Awareness Training?”.* [Online]   
Available at: https://www.techtarget.com/searchsecurity/definition/security-awareness-training?utm\_source=chatgpt.com

Wolford, B., n.d. *What is GDPR, the EU’s new data protection law?.* [Online]   
Available at: https://gdpr.eu/what-is-gdpr/?utm\_source=chatgpt.com

* <https://youtu.be/PzEWHH2v3TE?si=OwM4w-gGH7fE3iTg>
* <https://youtu.be/__0nZuG4sTw?si=3O0xy0ExGa9zYSzt>
* <https://youtu.be/3XqeCYnaSqc?si=3f8tHTuQFXgHNImL>
* <https://youtu.be/zteyEk9LADs?si=EgNBRjDObOo_70hj>
* <https://youtu.be/JiGRVIQ9rks?si=qTU4LS8voxZTshj7>
* Chat gpt for error correction and improved version of English
* Drae.io for drawing digrams
* Topbraid for q2
* Python
* Vs code
* <https://youtu.be/inWWhr5tnEA?si=gX8nh8IqFV05pnlJ>
* <https://youtu.be/z5nc9MDbvkw?si=fdRYXToe8-M7cBnt>
* MS-Excel Solver for q1
* <https://youtu.be/BrPRZLn-Zsk>
* <https://youtu.be/z5nc9MDbvkw?si=fdRYXToe8-M7cBnt>
* <https://youtu.be/TKiPTeARD9o?si=URHI604AZ-k4H-q5>
* <https://youtu.be/w5Xawyfrf0s>
* <https://youtu.be/TKiPTeARD9o?si=URHI604AZ-k4H-q5>
* [Introduction to A\*](https://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html)
* [Artificial Intelligence | Electrical Engineering and Computer Science | MIT OpenCourseWare](https://ocw.mit.edu/courses/6-034-artificial-intelligence-fall-2010/)
* [High capacity motors on-line diagnosis based on ultra wide band partial discharge detection | IEEE Conference Publication | IEEE Xplore](https://ieeexplore.ieee.org/document/1234567)
* [A Star algorithm | Example | Informed search | AI | Artificial intelligence | Lec-21 | Bhanu Priya](https://www.youtube.com/watch?v=PzEWHH2v3TE)
* [Latest News from Google Research Blog - Google Research](https://research.google/blog/)
* [High capacity motors on-line diagnosis based on ultra wide band partial discharge detection | IEEE Conference Publication | IEEE Xplore](https://ieeexplore.ieee.org/document/1234567)