# University of Global Village (UGV), Barishal


## Content of the Theory Course

## University Student (UGV) Format

## Program: Bachelor of Science in Computer Science Engineering (CSE)



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| Course Code | CSE-406 |
|---|---|
| **Name of Course Title** | **Artificial Intelligence Lab** |
| **Course Type** | **Core Course** |
| **Level** | **7th Semester** |
| **Academic Session** | **Summer 2024** |
| **Name(s) of AcademicCourse teacher(s)** | **Md. Riadul Islam**<br>**Assistant Professor**<br>**Mobile: 01842611852**<br>**E-mail: riadnwu@gmail.com** |
| **Consultation Hours:** | |

| Artificial Intelligence Lab Student (UGV) Format | |
|---|---|
| **Course Code:** CSE-406 | **Credits: 01** |
| **Exam Hours: 03** | **CIE Marks: 30** |
| **Course for 7th Semester,**<br>**Bachelor of Science in Computer Science Engineering (CSE)** | **SEE Marks: 20** |

## 1. Course Objectives:

- **Master basic search algorithms:** Students will gain a comprehensive understanding of fundamental search algorithms such as depth-first search, breadth-first search, and A* search. They will learn to apply these algorithms to solve a variety of problems efficiently.

- **Implement effective knowledge representation and reasoning techniques:** This objective focuses on equipping students with techniques to represent knowledge effectively within computational systems. They will explore various formalisms such as logic-based representations, semantic networks, and probabilistic models. Additionally, students will learn how to reason with this knowledge to draw meaningful conclusions.

- **Develop and deploy supervised and unsupervised learning algorithms:** Students will delve into the theory and practical implementation of both supervised (e.g., linear regression, support vector machines) and unsupervised (e.g., clustering, dimensionality reduction) learning algorithms. They will understand the principles behind these methods and gain proficiency in applying them to analyze datasets and make predictions.

- **Design and train neural networks proficiently:** This objective involves mastering the principles of neural network architectures (e.g., feedforward, convolutional, recurrent networks) and learning algorithms (e.g., backpropagation). Students will gain hands-on experience in designing, training, and optimizing neural networks for various tasks such as image recognition, natural language processing, and time series prediction.

- **Apply advanced natural language processing techniques:** Students will explore advanced techniques in natural language processing (NLP) such as sentiment analysis, named entity recognition, and machine translation. They will learn how to preprocess text data effectively, apply state-of-the-art models like transformers, and evaluate NLP systems' performance.



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

- **Implement reinforcement learning algorithms:** This objective focuses on understanding the principles of reinforcement learning (RL) algorithms such as Q-learning, policy gradients, and deep RL approaches. Students will learn how to apply RL to build agents that can learn optimal behavior through interaction with an environment, applicable to tasks such as game playing and robotics.

- **Gain hands-on experience in AI and machine learning projects:** Throughout the course, students will work on practical projects that integrate concepts learned across different modules. These projects will provide them with real-world experience in applying AI and machine learning techniques to solve complex problems and address specific challenges.

- **Explore cutting-edge topics and real-world applications in AI:** The course will cover advanced topics such as generative adversarial networks (GANs), explainable AI, ethical considerations in AI, and emerging trends in the field. Students will gain insights into how AI is shaping industries and society, preparing them to tackle future challenges and opportunities in the AI domain.

## 4. Course Learning Outcome (CLO) at the end of the course, the students will be able to-



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| | |
|---|---|
| **CLO1** | **Search Algorithms**<br><br>***Description:*** **Implement basic search algorithms like BFS, DFS, and A*.**<br><br>***Specific Learning Objectives****:*<br><br>• Understand the concepts of BFS and DFS.<br>• Implement BFS, DFS, and A* search algorithms.<br>• Apply these algorithms to solve search problems. |
| **CLO2** | **Knowledge Representation and Reasoning**<br><br>***Description:*** **Develop logic-based knowledge representation and reasoning systems.**<br><br>***Specific Learning Objectives:***<br><br>• Implement propositional and predicate logic.<br>• Develop inference rules and apply them.<br>• Use forward and backward chaining. |
| **CLO3** | **Supervised Learning** |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| | |
|---|---|
| | ***Description:*** **Implement and evaluate supervised learning algorithms.**<br><br>***Specific Learning Objectives:***<br><br>• Implement Linear and Logistic Regression.<br>• Develop Decision Trees and SVM models.<br>• Evaluate models using accuracy, precision, recall, and F1-score. |
| **CLO4** | **Unsupervised Learning**<br><br>***Description:*** **Implement clustering and dimensionality reduction techniques.**<br><br>***Specific Learning Objectives:***<br><br>• Implement K-means and Hierarchical clustering.<br>• Perform PCA and t-SNE.<br>• Visualize clustering and dimensionality reduction results. |
| **CLO5** | **Neural Networks and Deep Learning** |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| | |
|---|---|
| | ***Description:*** **Develop and train neural networks.**<br><br>***Specific Learning Objectives:***<br><br>• Build and train Feedforward, CNN, and RNN models.<br>• Use TensorFlow or PyTorch for model development.<br>• Evaluate and optimize neural networks. |
| **CLO6** | **Natural Language Processing (NLP)**<br><br>***Description:*** **Apply NLP techniques to text data.**<br><br>***Specific Learning Objectives:***<br><br>• Preprocess text data.<br>• Implement Sentiment Analysis and Named Entity Recognition (NER).<br>• Generate text using NLP models. |
| **CLO7** | **Reinforcement Learning** |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| | |
|---|---|
| | ***Description:*** **Implement reinforcement learning algorithms.**<br><br>***Specific Learning Objectives:***<br><br>• Develop reinforcement learning environments.<br>• Implement Q-learning and DQN.<br>• Apply Policy Gradient methods. |
| **CLO8** | **AI and Machine Learning Project Development**<br><br>***Description:*** **Apply AI techniques to real-world projects.**<br><br>***Specific Learning Objectives:***<br><br>• Define project scope and objectives.<br>• Collect and preprocess data.<br>• Implement and evaluate AI/ML algorithms. |
| **CLO9** | **Advanced Topics and Applications** |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| | |
|---|---|
| | ***Description:*** **Explore and implement advanced AI topics.**<br><br>***Specific Learning Objectives:***<br><br>• Implement GANs and Transfer Learning.<br>• Explore AI in robotics and autonomous systems. |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

## 5. Alignment to Program Learning Outcomes (PLOs)

| PLOs/CLOs | CLO1: Basic Search Algorithms | CLO2: Knowledge Representation and Reasoning | CLO3: Supervised and Unsupervised Learning | CLO4: Neural Networks | CLO5: Natural Language Processing | CLO6: Reinforcement Learning |
|---|---|---|---|---|---|---|
| PLO1 | 3 | 3 | 3 | 3 | 3 | 3 |
| PLO2 | 3 | 3 | 3 | 3 | 3 | 3 |
| PLO3 | 3 | 3 | 3 | 3 | 3 | 3 |
| PLO4 | 3 | 3 | 3 | 3 | 3 | 3 |
| PLO5 | 3 | 3 | 3 | 3 | 3 | 3 |

## 6. Mapping of CLOs to Program Learning Outcomes (PLOs)

**PLO1: Acquire a strong foundation in theoretical and practical aspects of computer science and engineering.**

a. CLO1: Basic Search Algorithms: Students will grasp foundational algorithms crucial for problem-solving in AI.
b. CLO2: Knowledge Representation and Reasoning: Students will learn formalisms essential for representing and reasoning about knowledge in AI.
c. CLO3: Supervised and Unsupervised Learning: Students will acquire practical knowledge in implementing machine learning algorithms, fundamental to AI applications.



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

d. CLO4: Neural Networks: Students will develop skills in designing and training neural networks, a cornerstone of modern AI.
e. CLO5: Natural Language Processing: Students will apply advanced techniques to process and understand human language, a critical aspect of AI.
f. CLO6: Reinforcement Learning: Students will implement algorithms that enable learning through interaction, essential for AI systems.

**PLO2: Apply knowledge of mathematics, science, and engineering principles to solve complex engineering problems.**

g. All CLOs contribute to applying theoretical knowledge to develop solutions in AI, from algorithms to practical applications like neural networks and natural language processing.

**PLO3: Design and conduct experiments, as well as analyze and interpret data to provide valid conclusions.**

h. CLO3 (Supervised and Unsupervised Learning) involves designing experiments to analyze data and draw conclusions, essential for evaluating AI models.
i. CLO6 (Reinforcement Learning) involves experimenting with AI agents in simulated environments to evaluate performance.



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

**PLO4: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.**

j. Group projects and assignments throughout the lab manual promote teamwork and collaborative problem-solving, preparing students for multidisciplinary AI applications.

**PLO5: Recognize the need for, and engage in, lifelong learning to continually update knowledge and skills in computer science and engineering.**

k. Exploration of advanced AI topics (CLO5 and CLO6) encourages students to stay updated with current AI trends and technologies, fostering a commitment to lifelong learning.



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

## 7. Topics to be covered/Content of the course

| Week | Topics | Specific Outcomes | Teaching Learning Strategy(s) | Assessment Strategy(s) | Alignment to CLO |
|---|---|---|---|---|---|
| 1. | Introduction to AI Tools and Environment (Part 1) | Set up and navigate Python and relevant libraries. | Hands-on practice, instructor-led setup | Feedback, observation | CLO1 |
| 2. | Acting Humanly: The Turing Test | Acting Humanly: The Turing Test | Practical exercises, group work | Q&A, practical assignments | CLO1 |
| 3. | Simple Rational Agent Chatbot in Python | Implement logic-based representation and inference rules. | Code implementation, peer review | Lab exercises, quizzes | CLO2 |
| 4. | BFS With MAZE | Implement Linear Regression, Logistic | Practical coding sessions, group work | Feedback, practical assignments | CLO3 |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| | | | | | |
|---|---|---|---|---|---|
| | | Regression, Decision Trees, SVM. | | | |
| 5. | DFS With MAZE | Implement K-means, Hierarchical clustering, PCA, t-SNE. | Hands-on exercises, instructor guidance | Observation, quizzes | CLO4 |
| 6. | N Queen Using DFS | Build and train neural networks using TensorFlow/ PyTorch. | Practical coding sessions, instructor guidance | Practical assignments, observation | CLO5 |
| 7. | 8 Puzzle Using BFS | Text preprocessing, Sentiment Analysis, NER, text generation. | Lecture, multimedia, interactive sessions | Feedback, Q&A, quizzes | CLO6 |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| | | | | | |
|---|---|---|---|---|---|
| 8. | A* Search | Develop and train reinforcement learning models. | Hands-on project work, peer collaboration | Observation, feedback | CLO7 |
| **Mid Term Examination** | | | | | |
| 9. | Conditional probability | Define project scope, collect and preprocess data, and implement algorithms. | Group discussions, instructor feedback | Observation, project planning | CLO8 |
| 10. | Genetic Algorithms To Solve The Traveling Salesman Problem (TSP) | Define project scope, collect and preprocess data, and implement algorithms. | Group discussions, instructor feedback | Observation, project planning | CLO8 |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| | | | | | |
|---|---|---|---|---|---|
| 11. | ACO : Ant Colony Optimization for TSP | Implement GANs, Transfer Learning, explore AI in robotics. | Hands-on project work, peer collaboration | Observation, feedback | CLO9 |
| 12. | ANN Forward pass and Back propagation | Implement advanced search algorithms and optimization techniques. | Practical exercises, group work | Q&A, practical assignments | CLO1 |
| 13. | Fuzzy logic control in automated car system | Implement Ensemble Methods (Random Forest, Gradient Boosting). | Practical coding sessions, group work | Feedback, practical assignments | CLO3 |
| 14. | Neural Networks and Deep Learning (Extended) | Implement advanced neural network architectures (ResNet, LSTM). | Practical coding sessions, instructor guidance | Practical assignments, observation | CLO5 |
| 15. | AI and Machine Learning Project Development (Extended) | Implement and refine project scope, collect and preprocess data, advanced algorithms. | Group discussions, instructor feedback | Observation, project planning | CLO8 |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

| | | | | | |
|---|---|---|---|---|---|
| 16. | Final Project Implementation | Implement and refine final AI/ML projects. | Hands-on project work, peer collaboration | Observation, feedback | All CL Os |
| **Final Term Examination** | | | | | |

8. **Teaching-Learning Strategy:**

- **Lectures:** Brief interactive sessions to explain AI concepts.
- **Practical Sessions:** Extensive hands-on practice to apply AI principles.
- **Group Discussions:** Collaborative activities to enhance understanding and problem-solving skills.
- **Assignments:** Regular tasks to reinforce learning and assess understanding.
- **Quizzes and Tests:** Periodic assessments to evaluate progress and mastery of concepts.

9. **Assessment Strategy:**

- **Continuous In-course Evaluation (CIE):** 30 marks
  - Lab Participation: 10 marks



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

    - Assignments: 10 marks
    - Quizzes: 10 marks
- **Final Project Evaluation:** 20 marks
    - Project Implementation: 10 marks
    - Project Presentation: 5 marks
    - Project Report: 5 marks
- **Total Marks:** 50

## 10. Instructional Materials and References:

- **Textbooks:**
    - "Artificial Intelligence: A Modern Approach" by Stuart Russell and Peter Norvig.
    - "Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville.

- **Reference Books:**
    - "Machine Learning: A Probabilistic Perspective" by Kevin P. Murphy.

- **Online Resources:**
    - AI Reference (http://www.aireference.com/)
    - Stack Overflow (https://stackoverflow.com/)
    - GeeksforGeeks (https://www.geeksforgeeks.org/)



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-1
# Set up and navigate Python and relevant libraries



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# 1. Objective

- Learn how to install and configure Python.
- Understand and navigate development environments (Anaconda, Jupyter Notebook, VS Code).
- Explore Python syntax and environment interaction.
- Install and use key Python libraries such as NumPy, Pandas, and Matplotlib.

---

# 2. Tools and Software

- **Python 3.10+**
- **Anaconda / Miniconda** (recommended)
- **Jupyter Notebook** or **VS Code**
- Internet access for installing additional libraries

---

# 3. Lab Tasks

1. **Python Installation and Setup**
    - Download and install Python or Anaconda.
    - Verify installation using the command:
    - `python --version`
    - Launch the environment (Jupyter Notebook or VS Code).
2. **Understanding Python Environment**



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

- Create and run a simple Python program (`print("Hello, Python!")`).
- Explore Python data types: integers, floats, strings, lists, tuples, and dictionaries.
- Practice basic operations and variable declarations.

3. **Installing and Importing Libraries**
   - Use pip to install common libraries:
   - `pip install numpy pandas matplotlib`
   - Import and test each library in Python:
   - `import numpy as np`
   - `import pandas as pd`
   - `import matplotlib.pyplot as plt`
4. **Hands-on with Core Libraries**
   - Use **NumPy** to create and manipulate arrays.
   - Use **Pandas** to create DataFrames and handle simple datasets.
   - Use **Matplotlib** to plot data visually.
5. **Practical Example**

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Create data
data = np.array([5, 10, 15, 20, 25])
df = pd.DataFrame(data, columns=['Values'])

# Display and plot
print(df)
plt.plot(df['Values'], marker='o')
plt.title('Sample Data Visualization')
plt.xlabel('Index')
plt.ylabel('Values')
plt.show()
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# 4. Evaluation Criteria

| Criteria | Marks | Description |
|---|---|---|
| Setup Verification | 20 | Successful Python and library installation |
| Program Execution | 30 | Ability to run and interpret basic scripts |
| Library Usage | 30 | Correct use of NumPy, Pandas, and Matplotlib |
| Report Submission | 20 | Summary with code and screenshots |

# 5. Expected Learning Outcomes

- Successfully install and configure Python.
- Operate in Jupyter Notebook or VS Code.
- Understand Python's basic syntax and environment.
- Use standard libraries for data handling and visualization.



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

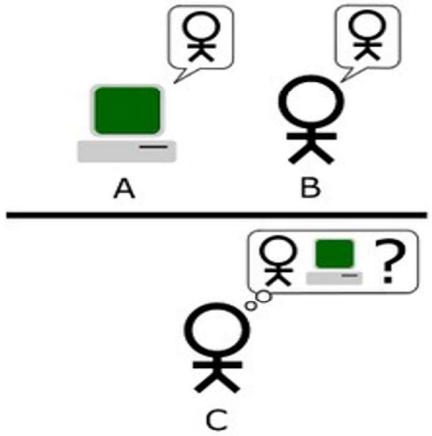
# WEEK-2
# Acting Humanly: The Turing Test



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Acting Humanly: The Turing Test

*http://en.wikipedia.org/wiki/Turing_test*




*Alan Turing*

*1912-1954*

- To be intelligent, a program should simply act like a human



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```python
# eliza_bot.py
import re
import random

reflections = {
    "i am": "you are",
    "i'm": "you are",
    "i": "you",
    "me": "you",
    "my": "your",
    "you are": "I am",
    "you": "I",
}
```

```python
pairs = [
    (r'.*\bhello\b.*', ["Hello. How can I help you today?", "Hi — what's on your mind?"]),
    (r'.*\b(sad|unhappy|depressed|upset)\b.*', ["I'm sorry you feel {0}. Do you want to talk about
    (r'.*\b(happy|great|good|fine)\b.*', ["That's good to hear! What made you {0}?"]),
    (r'.*\b(my|i am|i\'m) (.*)', ["Why do you say {1}?", "How long have you been {1}?"]),
    (r'.*\b(yes|no)\b.*', ["Why do you say {0}?", "Can you tell me more?"]),
    (r'.*\b(bye|goodbye|see you)\b.*', ["Goodbye. Take care!", "See you later!"]),
    (r'(.*)', ["Tell me more about {0}.", "Why do you think {0}?", "How does {0} make you feel?"])
]
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
def reflect(fragment):
    words = fragment.lower().split()
    for i,w in enumerate(words):
        if w in reflections:
            words[i] = reflections[w]
    return " ".join(words)

def respond(sentence):
    sentence = sentence.strip()
    for pattern, responses in pairs:
        m = re.match(pattern, sentence, re.IGNORECASE)
        if m:
            template = random.choice(responses)
            # insert captured groups if needed
            groups = [g for g in m.groups() if g is not None]
            if groups:
                # reflect first captured group
                return template.format(reflect(groups[-1]))
            else:
                return template.format(sentence)
    return "I see."
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```python
def chat():
    print("ELIZA-like bot. Type 'quit' to exit.")
    while True:
        user = input("You: ")
        if user.lower() in ['quit','exit','bye','goodbye']:
            print("Bot: Goodbye.")
            break
        print("Bot:", respond(user))

if __name__ == "__main__":
    chat()
```



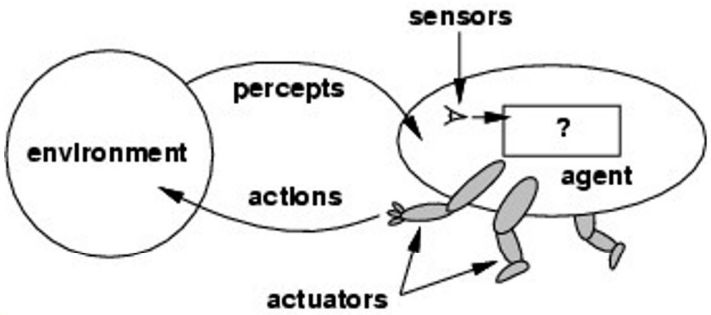
**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-3
# Simple Rational Agent Chatbot in Python



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Agents




- The agent function maps from percept histories to actions:
- $[f: \mathcal{P}^* \rightarrow \mathcal{A}]$
- Percept history is also known as percept sequence.



**How it works:**

1. **Perceive:** Takes user input.
2. **Think:** Checks its knowledge base for keywords.
3. **Act:** Responds with the best matching answer.
4. **Goal:** Give meaningful, rational replies to user questions.



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```python
class RationalAgent:
    def __init__(self):
        # Agent knowledge base
        self.knowledge = {
            "hello": "Hello! How can I assist you today?",
            "your name": "I am a Rational Agent designed to help you.",
            "what is ai": "AI stands for Artificial Intelligence – systems that can think and lear
            "what is rational agent": "A rational agent acts to achieve the best outcome based on
            "bye": "Goodbye! Have a great day."
        }

    def perceive(self, environment_input):
        """Perceive the environment (user input)."""
        return environment_input.lower()

    def think(self, perception):
        """Decide the best action based on knowledge."""
        for key in self.knowledge:
            if key in perception:
                return self.knowledge[key]
        return "I'm not sure about that. Could you please explain more?"

    def act(self, action):
        """Perform the chosen action (output message)."""
        print("Agent:", action)
```



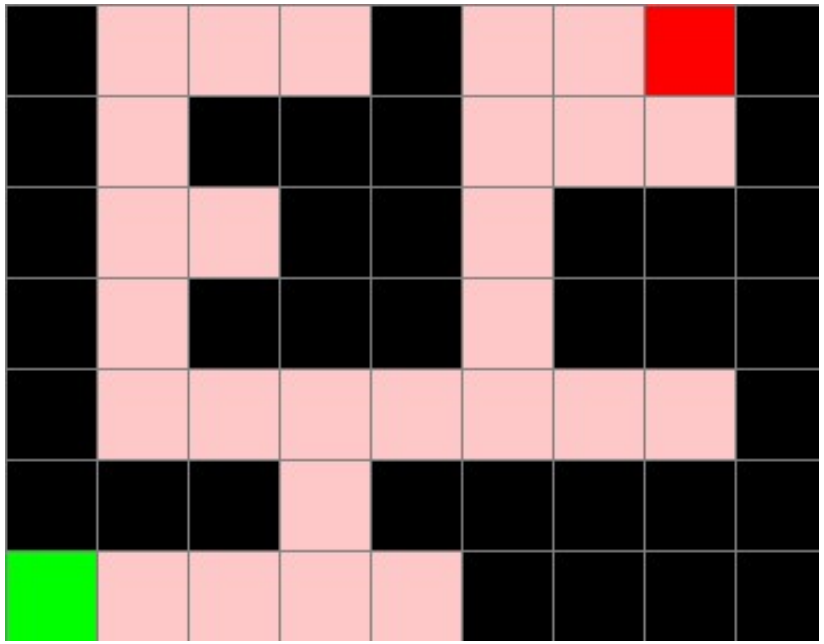
**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
    def run(self):
        """Main loop — perceive, think, act."""
        print("Rational Agent: Hello! Ask me something. (type 'bye' to exit)\n")
        while True:
            user_input = input("You: ")
            perception = self.perceive(user_input)
            response = self.think(perception)
            self.act(response)
            if "bye" in perception:
                break

# Run the agent
if __name__ == "__main__":
    agent = RationalAgent()
    agent.run()
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-4
# BFS With MAZE





**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
maze=open('maze.txt')
maze= [list(i) for i in maze.readlines()]
n=len(maze)
m=len(maze[0])
G=[0,8]
S=[6,0]
queue=[]
index=S
queue.append(index)

while index !=G:
    #print(queue)
    queue.pop()
    i=index[0]
    j=index[1]
    if i-1>-1:
        if maze[i-1][j] == ' ':
            queue.append([i-1,j])
            maze[i - 1][j] = '.'
        elif maze[i-1][j] == 'G':
            break
    if i + 1 < n - 1:
        if maze[i+1][j] == ' ':
            queue.append([i+1,j])
            maze[i + 1][j] = '.'
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
            elif maze[i+1][j] == 'G':
                break

        if j + 1 < m-1:
            if maze[i][j+1] == ' ':
                queue.append([i,j+1])
                maze[i][j+1] = '.'
            elif maze[i][j+1] == 'G':
                break
        if j-1>-1:
            if maze[i][j-1] == ' ':
                queue.append([i,j-1])
                maze[i][j-1] = '.'
            elif maze[i][j-1] == 'G':
                break
        print(queue[0])

        index=queue[-1]
for k in maze:
    for l in k:
        print(l, end='')
    print()

from PIL import Image, ImageDraw
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
# Size of each cell
cell_size = 40
rows = len(maze)
cols = max(len(row) for row in maze)

# Create a new image
img = Image.new("RGB", (cols * cell_size, rows * cell_size),
color="white")
draw = ImageDraw.Draw(img)

# Define colors for each character
colors = {
    '#': (0, 0, 0),         # Wall - Black
    '.': (255, 200, 200),   # Path - Light Gray
    'S': (0, 255, 0),       # Start - Green
    'G': (255, 0, 0),       # Goal - Red
    ' ': (255, 255, 255)    # Empty - White
}

# Draw each cell
for i, row in enumerate(maze):
    for j, ch in enumerate(row):
        color = colors.get(ch, (255, 255, 255))  # Default to white
        x0 = j * cell_size
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
            y0 = i * cell_size
            x1 = x0 + cell_size
            y1 = y0 + cell_size
            draw.rectangle([x0, y0, x1, y1], fill=color, outline="gray")

# Save or show the image
img.show()        # Show in default viewer
# img.save("maze.png")  # Or save to a file
```



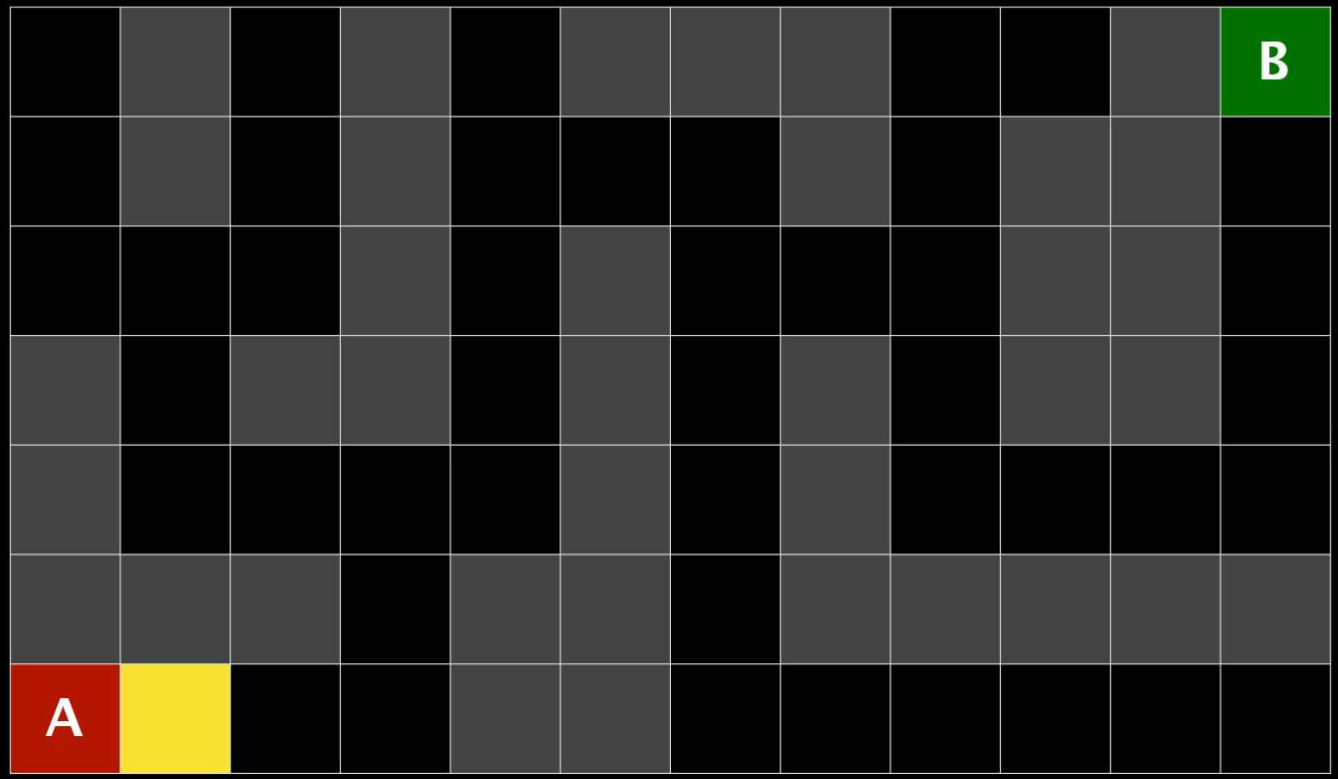
**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-5
# DFS With MAZE



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Depth-First Search






**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
maze=open('maze.txt')
maze= [list(i) for i in maze.readlines()]
n=len(maze)
m=len(maze[0])
G=[0,4]
S=[6,0]
stack=[]
index=S
stack.append(index)

while index !=G:
    #print(stack)
    stack.pop(-1)
    i=index[0]
    j=index[1]
    if i-1>-1:
        if maze[i-1][j] == ' ':
            stack.append([i-1,j])
            maze[i - 1][j] = '.'
        elif maze[i-1][j] == 'G':
            break
    if i + 1 < n - 1:
        if maze[i+1][j] == ' ':
            stack.append([i+1,j])
            maze[i + 1][j] = '.'
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
            elif maze[i+1][j] == 'G':
                break
        if j + 1 < m-1:
            if maze[i][j+1] == ' ':
                stack.append([i,j+1])
                maze[i][j+1] = '.'
            elif maze[i][j+1] == 'G':
                break
        if j-1>-1:
            if maze[i][j-1] == ' ':
                stack.append([i,j-1])
                maze[i][j-1] = '.'
            elif maze[i][j-1] == 'G':
                break


        index=stack[len(stack)-1]
    for k in maze:
        for l in k:
            print(l, end='')
        print()

from PIL import Image, ImageDraw


# Size of each cell
cell_size = 40
rows = len(maze)
cols = max(len(row) for row in maze)
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
# Create a new image
img = Image.new("RGB", (cols * cell_size, rows * cell_size), color="white")
draw = ImageDraw.Draw(img)

# Define colors for each character
colors = {
    '#': (0, 0, 0),         # Wall - Black
    '.': (255, 200, 200),   # Path - Light Gray
    'S': (0, 255, 0),       # Start - Green
    'G': (255, 0, 0),       # Goal - Red
    ' ': (255, 255, 255)    # Empty - White
}

# Draw each cell
for i, row in enumerate(maze):
    for j, ch in enumerate(row):
        color = colors.get(ch, (255, 255, 255))  # Default to white
        x0 = j * cell_size
        y0 = i * cell_size
        x1 = x0 + cell_size
        y1 = y0 + cell_size
        draw.rectangle([x0, y0, x1, y1], fill=color, outline="gray")

# Save or show the image
img.show()        # Show in default viewer
# img.save("maze.png")  # Or save to a file
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-6
# N Queen Using DFS

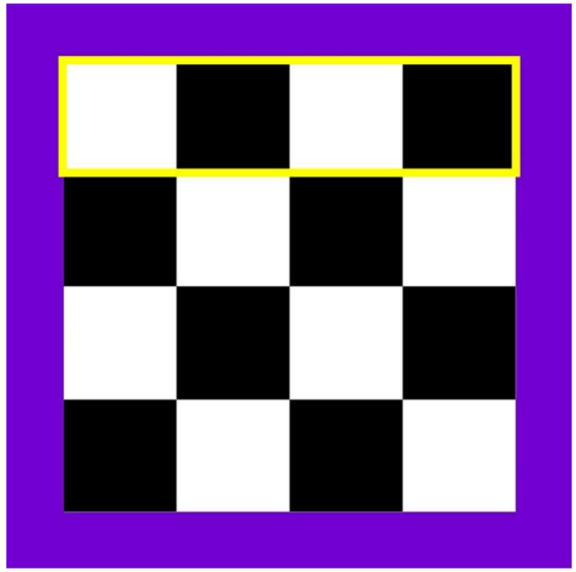


**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# N-Queens: Example

- Let's illustrate the backtracking algorithm with a mathematical example for N = 4. The chessboard as an N × N matrix.
- Now, we start placing queens using the backtracking algorithm:

1. Set Q1 to 1st row of 1st column.
2. Set Q2 to 2nd row of 3rd column.
3. Q3 has no place to 3rd row.
4. Backtrack to Q2 and set 2nd row of 4th column.
5. Set Q3 to 3rd row of 2nd column.





**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
import numpy as np
from numpy import bool


def q_pos(index,type):
    x=index[0]
    y=index[1]
    if type==1:
        bord[x][y]=-1
    else:
        bord[x][y] = 0
    q_over=((1,0),(0,1),(-1,0),(0,-1),(-1,-1),(-1,1),(1,1),(1,-1))
    for a in q_over:
        i=a[0]+x
        j=a[1]+y
        while(i>=0 and j>=0 and i<n and j<n and bord[i][j]!=-1):
            if type==1:
                bord[i][j]= bord[i][j]+1
            else:
                bord[i][j] = bord[i][j] - 1
            i =i+a[0]
            j =j+a[1]

def n_quee(num_queen):
    if num_queen==n:
        return 1
    free_index=np.where(bord[num_queen]==0)
    free_index=free_index[0].tolist()
    #print(free_index)
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
    while len(free_index)!=0:
        i=free_index[0]
        #print("Q",num_queen)
        free_index=np.delete(free_index,0)
        q_pos((num_queen,i), 1)
        #print(bord)
        flag=n_quee(num_queen+1)
        if flag==1:
            return 1
        q_pos((num_queen, i), 0)

import numpy as np
from PIL import Image, ImageDraw, ImageFont

def display(board):
    n = board.shape[0]
    cell_size = 80
    img_size = n * cell_size

    # Create blank image
    img = Image.new("RGB", (img_size, img_size), "white")
    draw = ImageDraw.Draw(img)

    # Load font
    try:
        font = ImageFont.truetype("arial.ttf", 30)
    except:
        font = ImageFont.load_default()
```

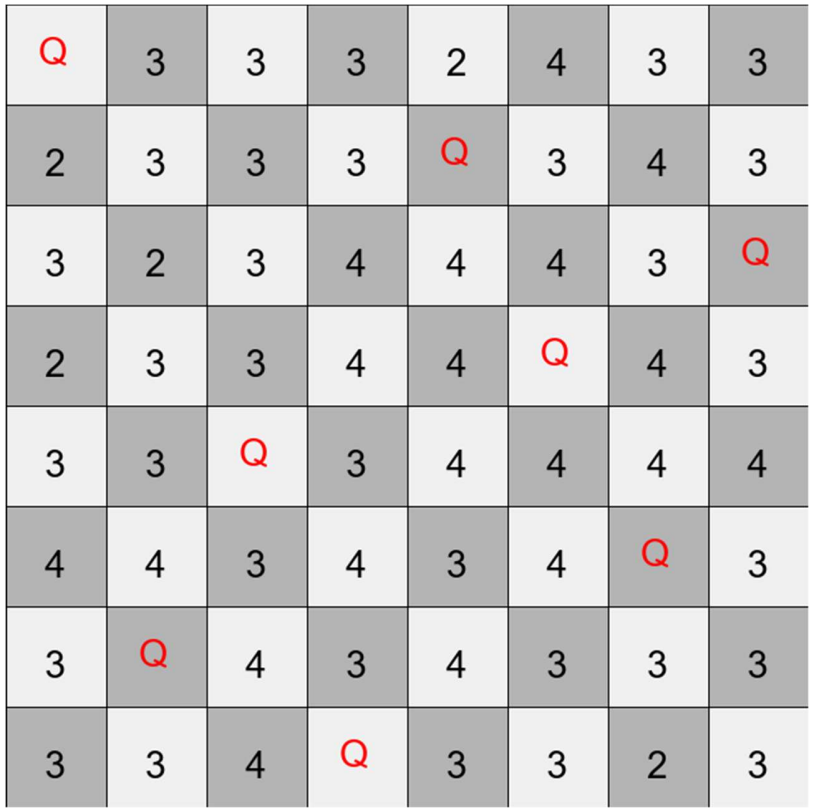


**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
    for i in range(n):
        for j in range(n):
            x0, y0 = j * cell_size, i * cell_size
            x1, y1 = x0 + cell_size, y0 + cell_size

            # Chessboard pattern
            fill = (240, 240, 240) if (i + j) % 2 == 0 else (180, 180, 180)
            draw.rectangle([x0, y0, x1, y1], fill=fill, outline="black")

            value = board[i, j]
            if value == -1:  # queen
                draw.text((x0 + cell_size//3, y0 + cell_size//4), "Q", fill="red",
font=font)
            elif value > 0:  # overlap number
                # Use textbbox for proper centering
                bbox = draw.textbbox((0, 0), str(value), font=font)
                w, h = bbox[2] - bbox[0], bbox[3] - bbox[1]
                draw.text((x0 + (cell_size - w)//2, y0 + (cell_size - h)//2),
                          str(value), fill="black", font=font)

    img.show()


n=8
bord=np.zeros((n,n),dtype=int)
num_queen=0
n_quee(num_queen)
display(bord)
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha





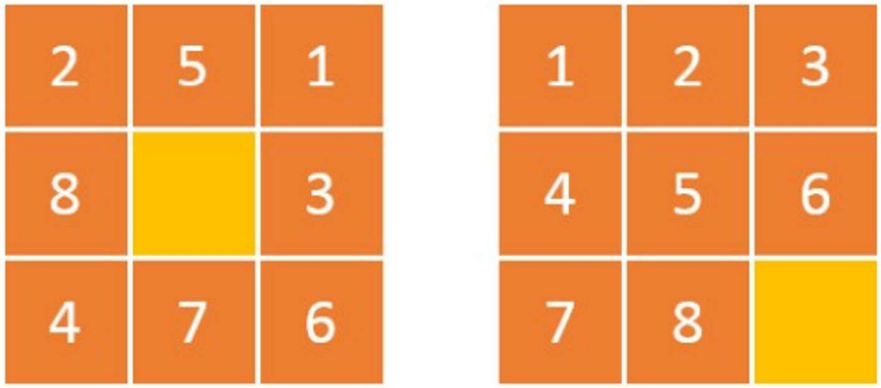

**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-7
# 8 Puzzle Using BFS



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Solve this Eight Puzzle using BFS






**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
import copy
def find_zero(p_state):
    x=0
    while True:
        if 0 in p_state[x]:
            y=p_state[x].index(0)
            return x,y
        else:
            x=x+1

def position_check():
    while queue[0]!= t_state:
        print(queue[0])
        print('@@@@@@@@@@@@@@@@')
        index=[[1,0],[-1,0],[0,1],[0,-1]]

        x, y =find_zero(queue[0])
        for a in index:
            i=a[0]+x
            j=a[1]+y
            if i>=0 and j>=0 and i<n and j<n:
                temp_state =copy.deepcopy(queue[0])
                temp_state[x][y] =temp_state[i][j]
                temp_state[i][j] =0
                print(temp_state)

                if temp_state in pop_list:
                    a=1
                elif temp_state == t_state:
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
                    print(temp_state)
                    print("result Found")
                    return 0
                else:
                    queue.append(temp_state)
                    pop_list.append(temp_state)
        queue.pop(0)
        print('-------------')
    print(queue[0])
```

```
p_state=[[1,2,3],[4,5,6],[7,8,0]]
t_state=[[1,3,0],[6,2,4],[7,5,8]]
print(t_state)
print('++++++++++++++++++++++++++')
n=len(p_state)
queue=[]
pop_list=[]
queue.append(p_state)
pop_list.append(p_state)
position_check()
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-8
# 8 Puzzle Using BFS



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# A* search

search algorithm that expands node with lowest value of $g(n) + h(n)$

$g(n)$ = cost to reach node
$h(n)$ = estimated cost to goal



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# A* Search

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | B |
| | 11 | | | | | | | | | | 1 |
| | 12 | | 10 | 9 | 8 | 7 | 6 | 5 | 4 | | 2 |
| | 13 | | 11 | | | | | | 5 | | 3 |
| | 14 | 13 | 12 | | 10 | 9 | 8 | 7 | 6 | | 4 |
| | | | 13 | | 11 | | | | | | 5 |
| A | 16 | 15 | 14 | | 12 | 11 | 10 | 9 | 8 | 7 | 6 |



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```python
import heapq

# Define the maze: 0 = free space, 1 = obstacle
maze = [
    [0, 1, 0, 0, 0, 0],
    [0, 1, 0, 1, 1, 0],
    [0, 0, 0, 1, 0, 0],
    [0, 1, 1, 1, 0, 1],
    [0, 0, 0, 0, 0, 0]
]

start = (0, 0)  # Starting position
end = (4, 5)    # Goal position

# Heuristic: Manhattan distance
def heuristic(a, b):
    return abs(a[0] - b[0]) + abs(a[1] - b[1])

# A* algorithm
def astar(maze, start, end):
    rows, cols = len(maze), len(maze[0])
    open_set = []
    heapq.heappush(open_set, (0 + heuristic(st
, end), 0, start, [start]))
    visited = set()
```

```
    while open_set:
        _, cost, current, path = heapq.heappop(open_set)

        if current in visited:
            continue
        visited.add(current)

        if current == end:
            return path

        # Explore neighbors (up, down, left, right)
        neighbors = [(0,1),(1,0),(0,-1),(-1,0)]
        for dx, dy in neighbors:
            nx, ny = current[0] + dx, current[1] + dy
            if 0 <= nx < rows and 0 <= ny < cols and maze[nx][ny] == 0:
                if (nx, ny) not in visited:
                    new_cost = cost + 1
                    heapq.heappush(open_set, (new_cost + heuristic((nx, ny), end), new_cost, (nx,

    return None  # No path found
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```python
# Run A*
path = astar(maze, start, end)

if path:
    print("Path found:")
    for step in path:
        print(step)
else:
    print("No path found.")
```

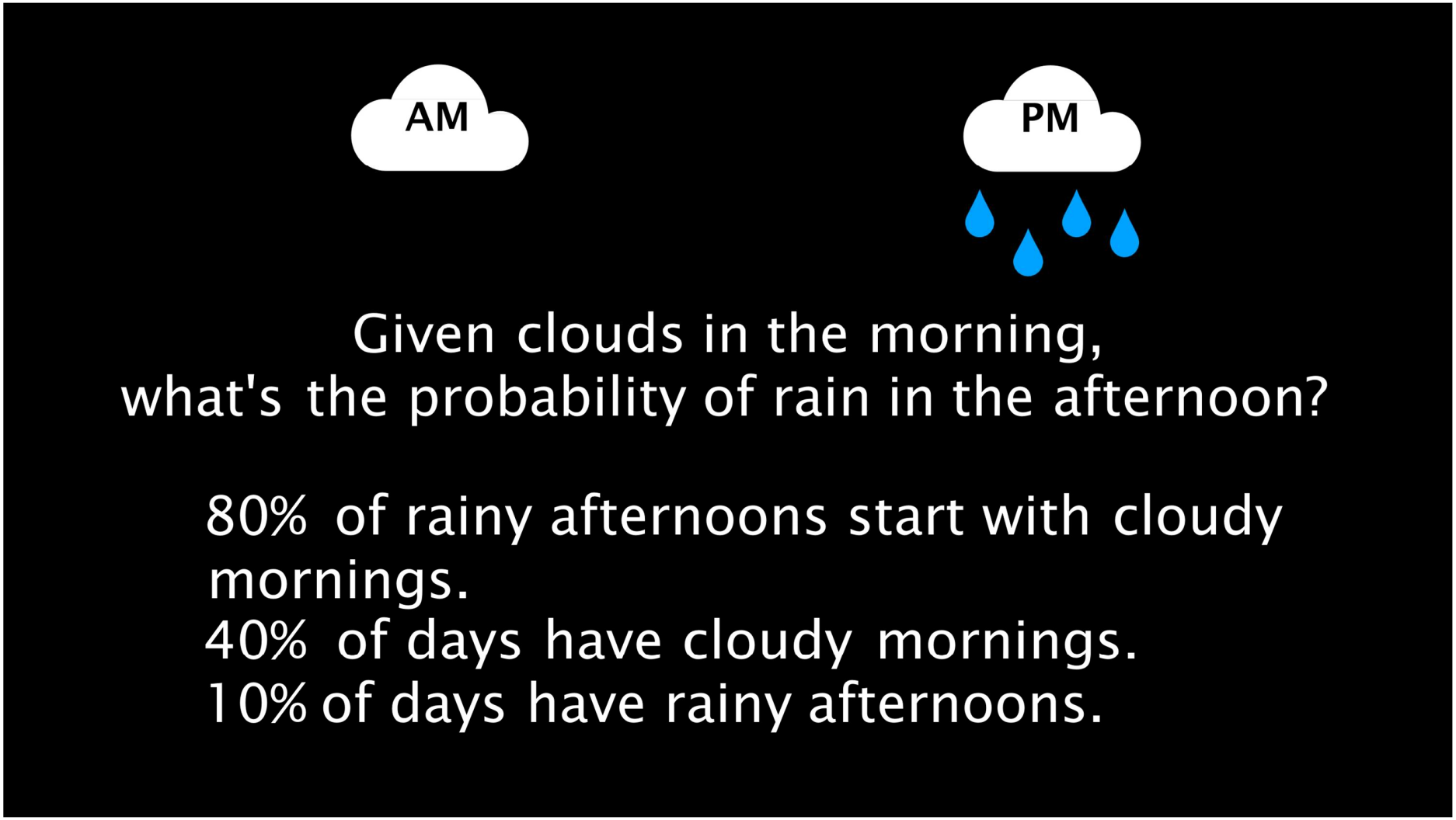


**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-9
# Conditional Probability

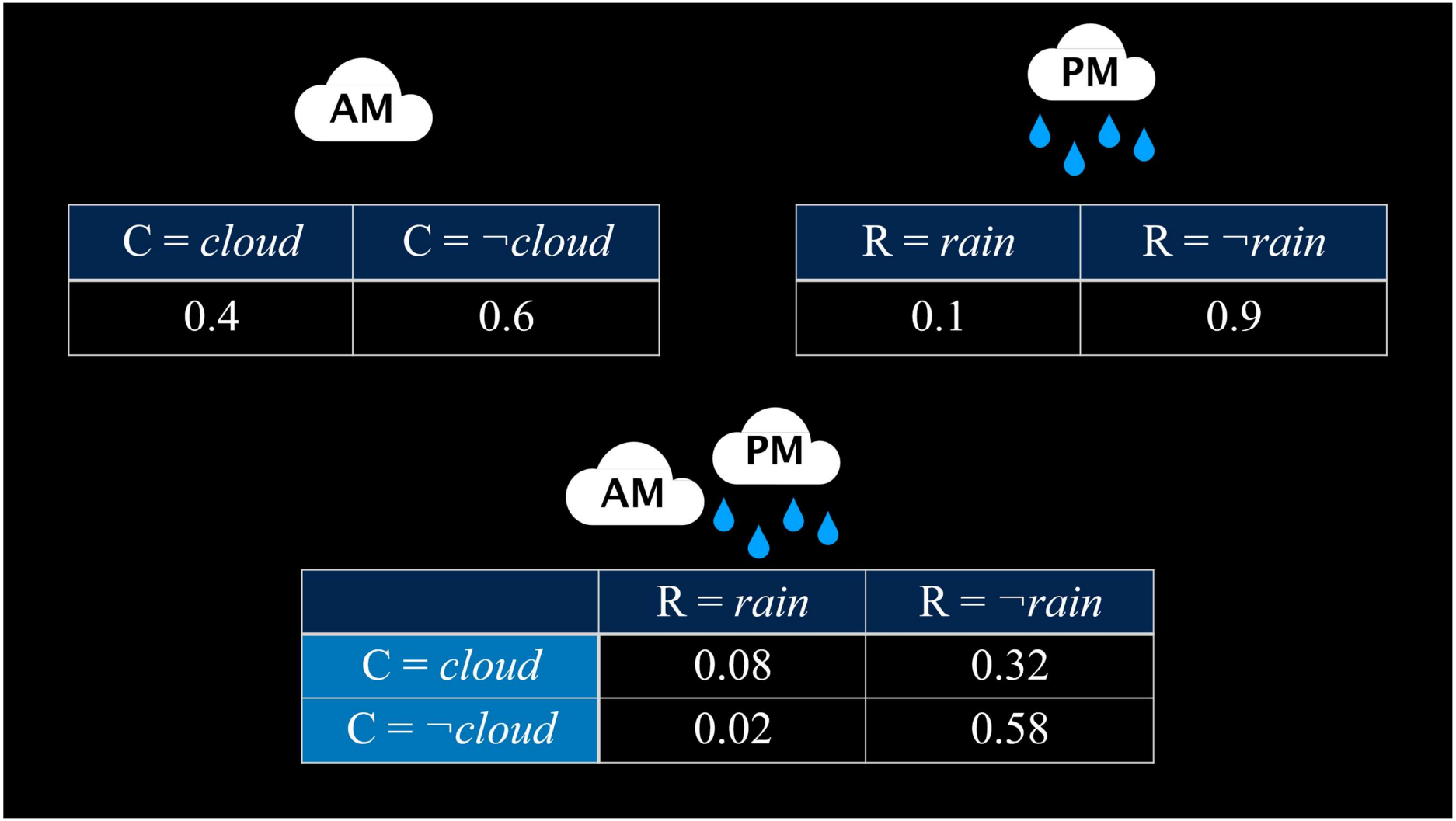


**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha






**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha






**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

$$\mathbf{P}(C \mid rain)$$

$$\mathbf{P}(C \mid rain) = \frac{\mathbf{P}(C, rain)}{P(rain)} = \alpha \mathbf{P}(C, rain)$$

$$= \alpha \langle 0.08, 0.02 \rangle = \langle 0.8, 0.2 \rangle$$

| | R = *rain* | R = ¬*rain* |
|---|---|---|
| C = *cloud* | 0.08 | 0.32 |
| C = ¬*cloud* | 0.02 | 0.58 |

**Given data:**

- $\boldsymbol{P}$**(Cloudy Morning | Rainy Afternoon) = 0.8**



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

- $P(\text{Cloudy Morning}) = 0.4$
- $P(\text{Rainy Afternoon}) = 0.1$

---

**We need to find:**
**Usually, such problems ask for $P(\text{Rainy Afternoon} \mid \text{Cloudy Morning})$—the probability that the afternoon is rainy given the morning is cloudy.**
**We can use Bayes' theorem:**

$$P(R \mid C) = \frac{P(C \mid R) \times P(R)}{P(C)}$$

---

**Substitute the values:**

$$P(R \mid C) = \frac{0.8 \times 0.1}{0.4} = 0.2$$

**So,**
**The probability of a rainy afternoon given a cloudy morning is 0.2 (or 20%).**

---



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```python
# Given probabilities
P_cloudy_given_rainy = 0.8
P_cloudy = 0.4
P_rainy = 0.1

# Using Bayes' theorem
P_rainy_given_cloudy = (P_cloudy_given_rainy * P_rainy) / P_cloudy

print(f"Probability of rainy afternoon given cloudy morning: {P_rainy_given_cloudy:.2f}")
```

✅ **Output:**
**Probability of rainy afternoon given cloudy morning: 0.20**



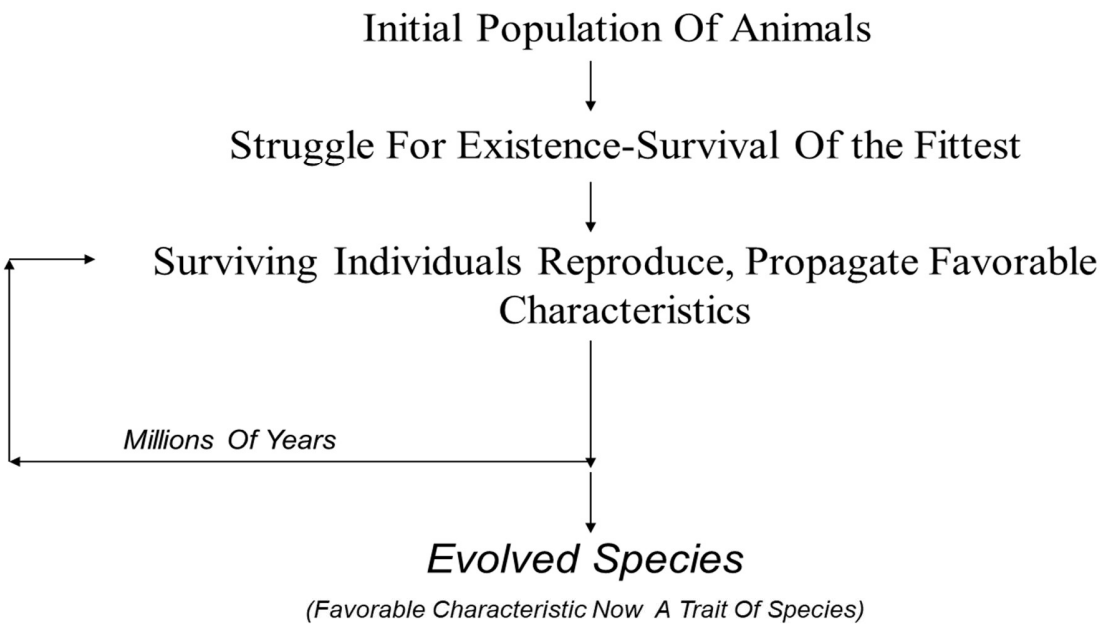
**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-10
# Genetic Algorithms To Solve The Traveling Salesman Problem (TSP)



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Evolution Through Natural Selection






• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Simple Genetic Algorithm

```
Simple_Genetic_Algorithm()
      {
      Initialize the Population;
      Calculate Fitness Function;

      While(Fitness Value != Optimal Value)
            {
            Selection;//Natural Selection, Survival
  Of Fittest
            Crossover;//Reproduction, Propagate
  favorable characteristics

            Mutation;//Mutation
            Calculate Fitness Function;
            }
      }
```

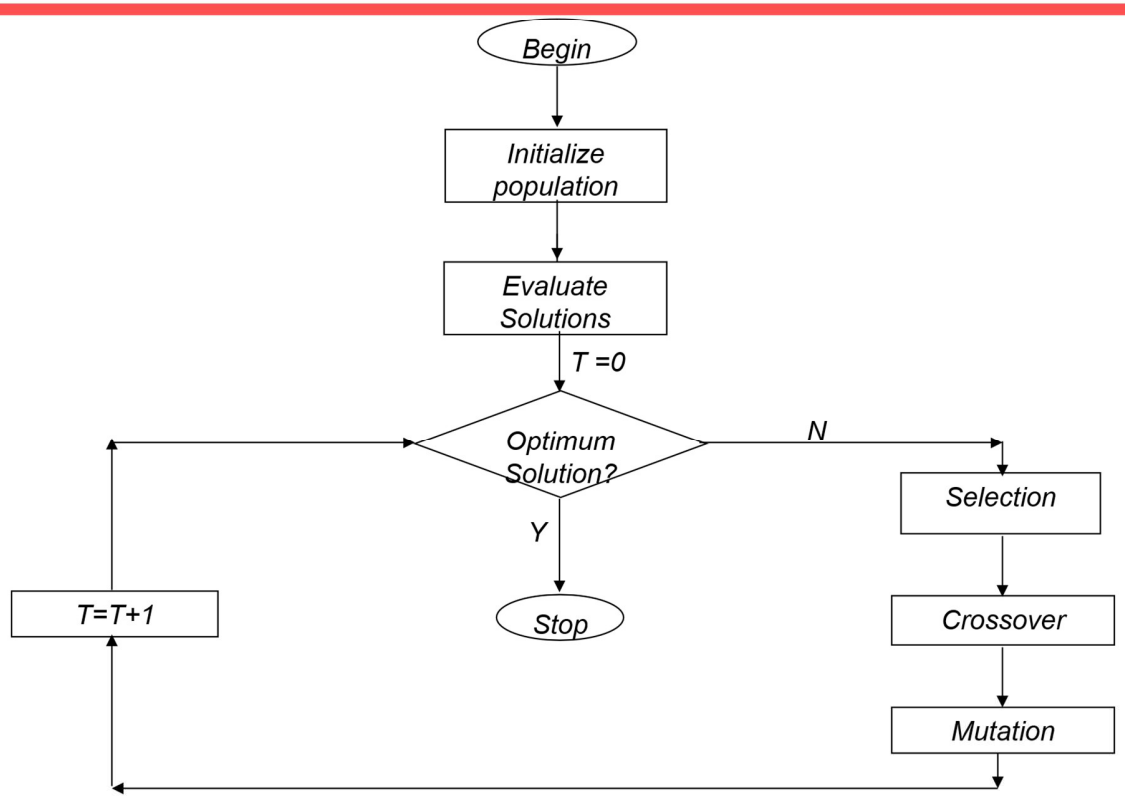
78





**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Working Mechanism Of GAs




77

• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Problem Setup: 5 Cities

- Cities: A, B, C, D, E
- Distance Matrix:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | 2 | 9 | 10 | 7 |
| B | 2 | 0 | 6 | 4 | 3 |
| C | 9 | 6 | 0 | 8 | 5 |
| D | 10 | 4 | 8 | 0 | 6 |
| E | 7 | 3 | 5 | 6 | 0 |

• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Step 1: Initial Population

- Population size = 4 (example):
  - Individual 1: A → B → C → D → E → A
  - Individual 2: A → C → E → B → D → A
  - Individual 3: A → D → E → C → B → A
  - Individual 4: A → E → B → D → C → A

• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Step 2: Fitness Calculation

- Fitness = 1 / total distance
- Example:
  - Individual 1: Total distance = 29, Fitness = 1 / 29 ≈ 0.0345
  - Individual 2: Total distance = 31, Fitness = 1 / 31 ≈ 0.0322
  - Individual 3: Total distance = 29, Fitness = 1 / 29 ≈ 0.0345
  - Individual 4: Total distance = 31, Fitness = 1 / 31 ≈ 0.0322
- Total fitness
  - = (0.0345+0.0322+0.0345+ 0.0322)
  - =0.1334

• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Step 3: Selection (Roulette Wheel)

- Total fitness = 0.1334
- Selection probabilities:
  - Individual 1: (0.0345 ÷ 0.1334) =0.2586
  - Individual 2: (0.0322 ÷ 0.1334) = 0.2412
  - Individual 3: (0.0345 ÷ 0.1334) = 0.2586
  - Individual 4: (0.0322 ÷ 0.1334) = 0.2412
- Selection: Randomly pick parents based on fitness probabilities.

• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV

```
#   A B C D E
#   0 1 2 3 4
```

```
import random
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
def population_gen(n):
    population=[]
    for i in range(n):
        l = [0, 1, 2, 3, 4]
        random.shuffle(l)
        population.append(l)
    return population
def fitness_cal():
    fitness=[0]*n
    for i in range(n):
        for j in range(len(l)-1):
            fitness[i]=fitness[i]+
g[population[i][j]][population[i][j+1]]
        fitness[i]=1/fitness[i]
    print(fitness)
    total_fitness = sum(fitness)
    print(total_fitness)
    for i in range(n):
        fitness[i]=fitness[i]/total_fitness
    return fitness

def selection(population,fitness):
    parent=random.choices(population,weights=fitness,k=2)
    print(parent)
    return parent

def corssover(point):
    child=parent[0][:point]+parent[1][point:]
    child=list(child)
    print(child)
    miss_val=[]
    rep_index=[]
    for i in range(len(l)):
        if child.count(i)==0:
            miss_val.append(i)
        elif child.count(i)>1:
            rep_index.append(child.index(i))
    print(miss_val)
    print(rep_index)
    for i in range(len(rep_index)):
        child[rep_index[i]]=miss_val[i]
    print(child)
    return child
def mutation(child):
    random_index= random.sample(l, 2)
    print(random_index[0])
    print(random_index[1])
    temp1=child[random_index[0]]
    temp2=child[random_index[1]]
    child[random_index[0]]=temp2
    child[random_index[1]]=temp1
    print(child)
    return child
g=[[0,3,4,5,6],
   [2,0,5,8,9],
   [4,5,0,3,1],
   [2,3,5,0,4],
   [7,2,3,4,0],
   ]
print(g)

l=[0,1,2,3,4]
n=5
print(l)
population=population_gen(n)
print(population)
m=10
for i in range(m):
    fitness=fitness_cal()
    print(fitness)
    parent=selection(population,fitness)
    point=3
    child=corssover(point)
    child=mutation(child)
    population[0]=child
```



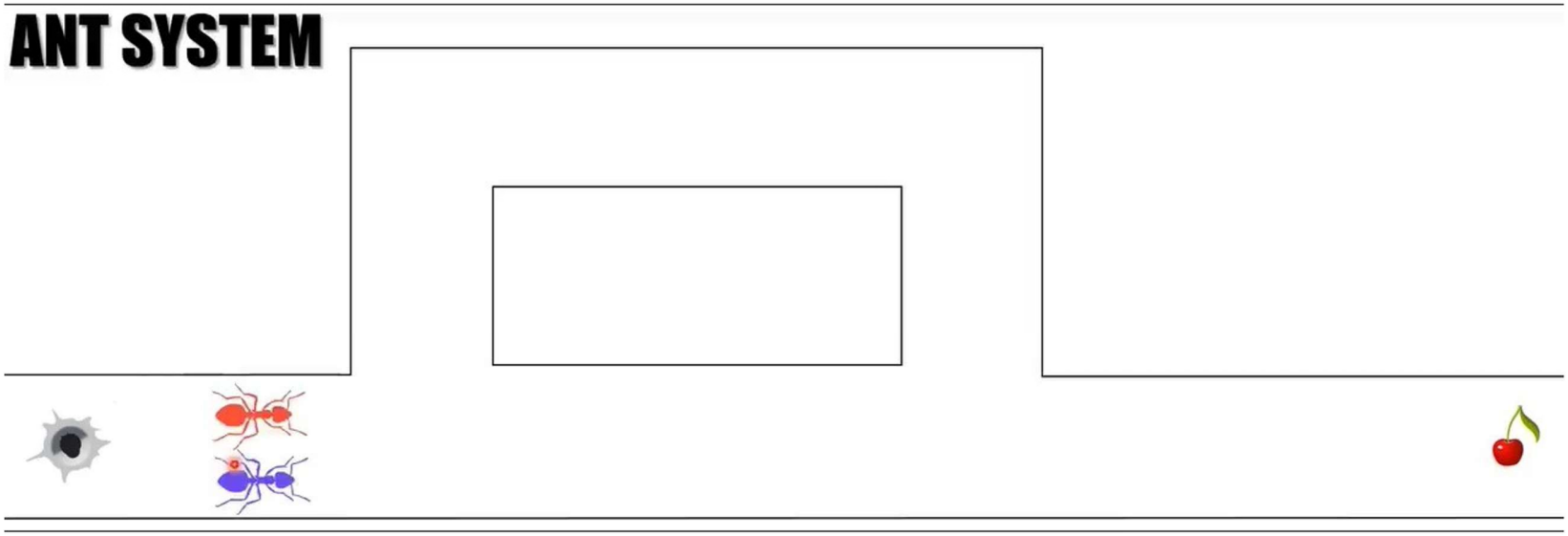
**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-10
# ACO Algorithm for TSP



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

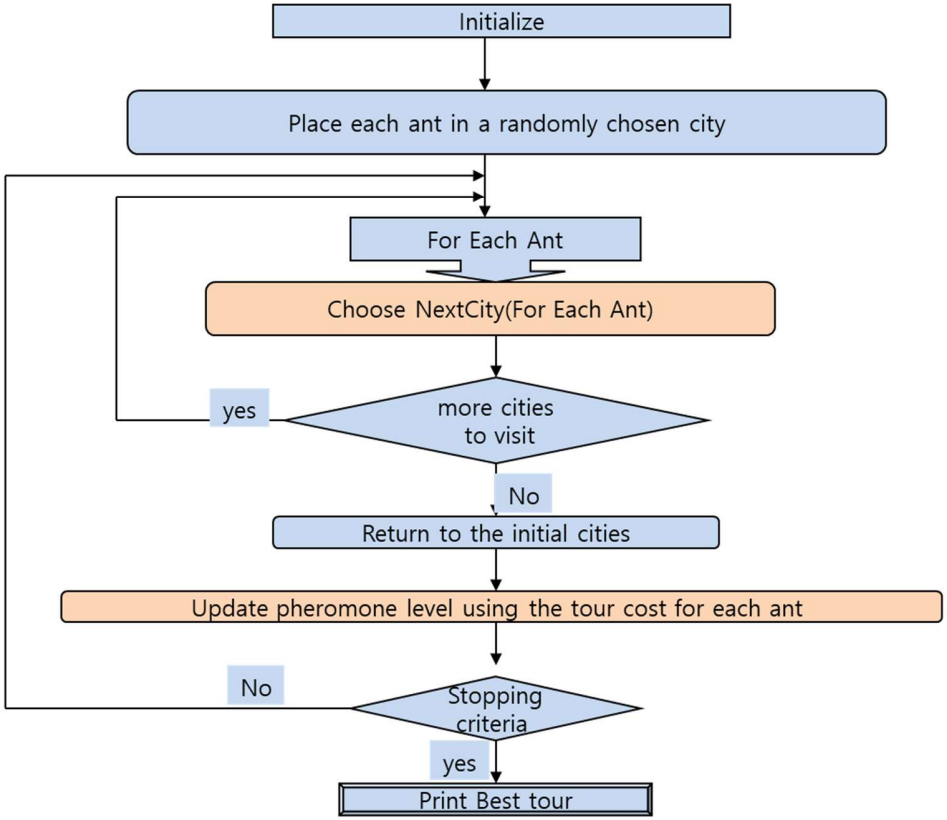
# Ant Colony Optimization algorithm






**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# ACO Algorithm for TSP





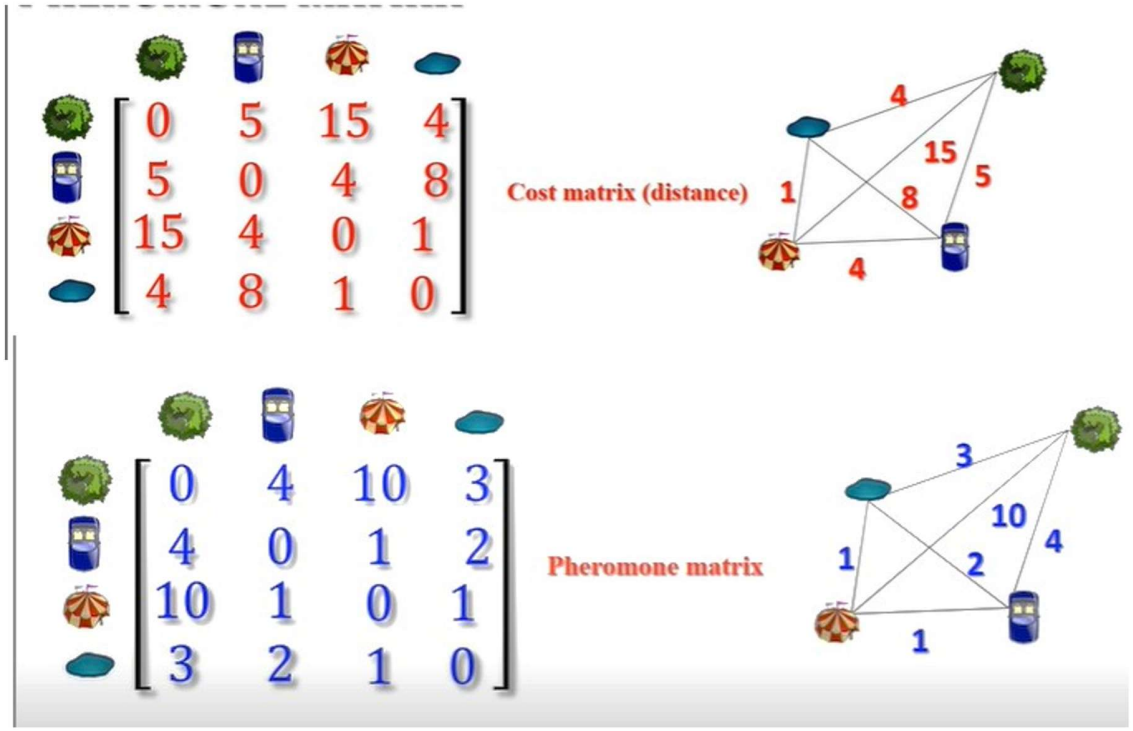

**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
Loop
   Randomly position m artificial ants on n cities
   For city=1 to n
      For ant=1 to m
        {Each ant builds a solution by adding one city after
        the other}
         Select probabilistically the next city according to
                        exploration and exploitation mechanism
         Apply the local trail updating rule
      End for
   End for
   Apply the global trail updating rule using the best ant
Until End_condition
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

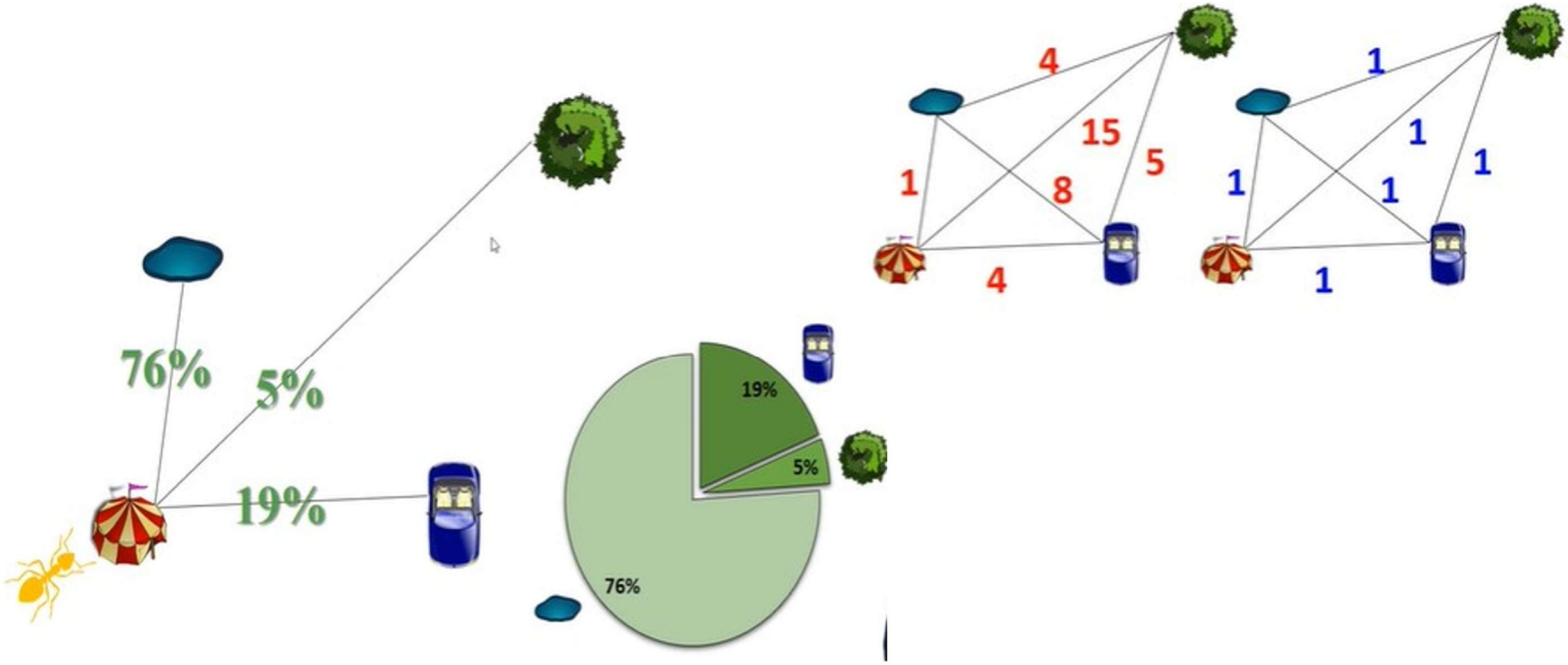
# Ant Colony Optimization algorithm






**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Ant Colony Optimization algorithm

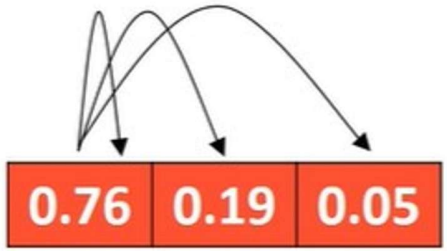


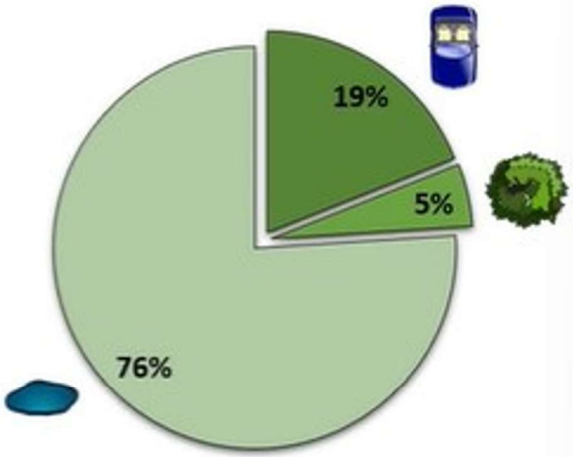



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Ant Colony Optimization algorithm

**Probabilistic**




**Cumulative sum**

| 1 | 0.24 | 0.05 |
|---|---|---|

**A random number (r) in [0,1]**

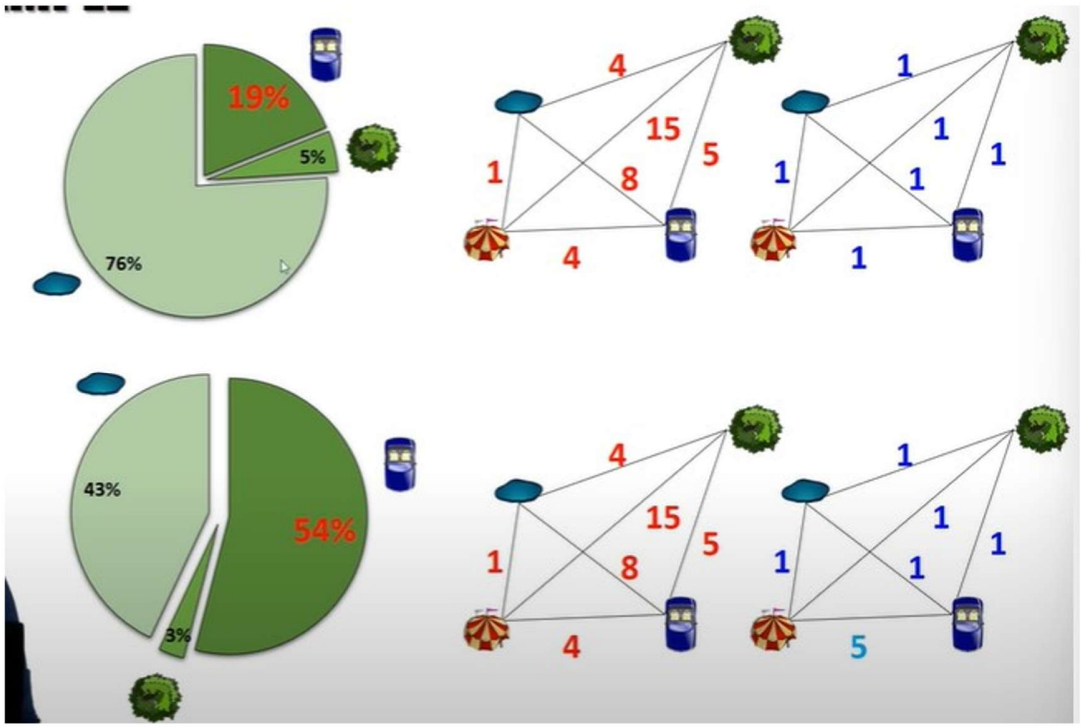
$$\begin{cases} 0.24 < r \leq 1.00 \\ 0.05 < r \leq 0.24 \\ 0.00 \leq r \leq 0.05 \end{cases}$$

154



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Ant Colony Optimization algorithm




• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV

156

**ACO CODE with Python**



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
import random
def population_gen(n):
    population=[]
    for i in range(n):
        temp_l = [0, 1, 2, 3, 4]
        random.shuffle(temp_l)
        population.append(temp_l)
    print(population)
    return population
def phermeon_cal():
    fitness=[0]*n

    for i in range(n):
        for j in range(len(l)-1):
            index1=population[i][j]
            index2 = population[i][j+1]
            fitness[i]=fitness[i]+g[index1][index2]
        fitness[i] =1/fitness[i]
    print(fitness)
    for i in range(n):
        for j in range(len(l) - 1):
            index1 = population[i][j]
            index2 = population[i][j + 1]
            phermeon[index1][index2]=
phermeon[index1][index2]+ fitness[i]
    print(phermeon)
    return phermeon
def new_ant_gen():
    for i in range (n):
        old_population=population[i]
        population[i]=[]
        present_city = old_population[0]
        population[i].append(present_city)
        old_population.remove(present_city)
        print(old_population)
        while len(old_population)>0:
            temp_value=[]
            for j in old_population:

temp_value.append(phermeon[present_city][j]*(1/g[present
_city][j]))
            total_temp=sum(temp_value)
            for j in range (len(temp_value)):
                temp_value[j]=temp_value[j]/total_temp
            present_city =
random.choices(old_population, weights=temp_value)[0]
            old_population.remove(present_city)
            population[i].append(int(present_city))
        print(population[i])
        print('----------------')

def best_root():
    fitness = [0] * n
    for i in range(n):
        for j in range(len(l)-1):
            index1=population[i][j]
            index2 = population[i][j+1]
            fitness[i]=fitness[i]+g[index1][index2]
    print(min(fitness))
    index_min=fitness.index(min(fitness))
    print(population[index_min])


g=[[0,30,4,5,60],
   [2,0,5,8,9],
   [4,5,0,30,19],
   [20,3,5,0,47],
   [70,2,3,4,0],
   ]
print(g)
#  A B C D E
city=5
phermeon=[[0]*city]*city
l=[0,1,2,3,4]
n=4
population=population_gen(n)
for i in range(50):
    phermeon=phermeon_cal()
    new_ant_gen()
    best_root()
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-11
# Artificial Neural Networks (ANNs): Forward Pass and Backpropagation



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

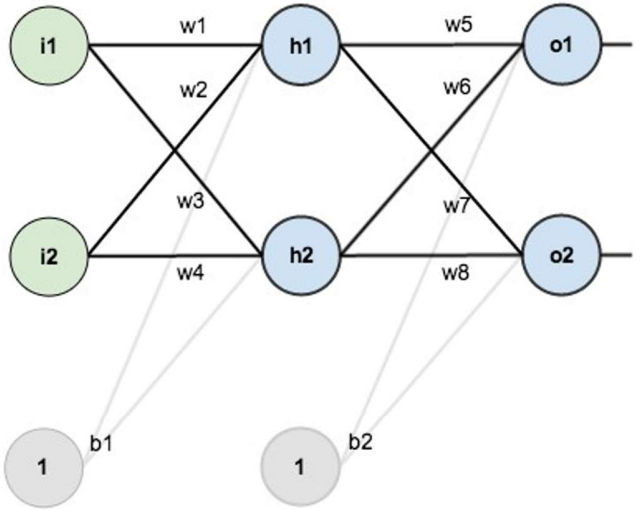
# Forward pass and Backpropagation

## *The Forward Pass*

- $net_{h1} = (w_1 \times i_1) + (w_2 \times i_2) + (b_1 \times 1)$
- $net_{h2} = (w_3 \times i_1) + (w_4 \times i_2) + (b_1 \times 1)$

*Activation Functions*

- $out_{h1} = \frac{1}{1+e^{-net_{h1}}}$
- $out_{h2} = \frac{1}{1+e^{-net_{h2}}}$





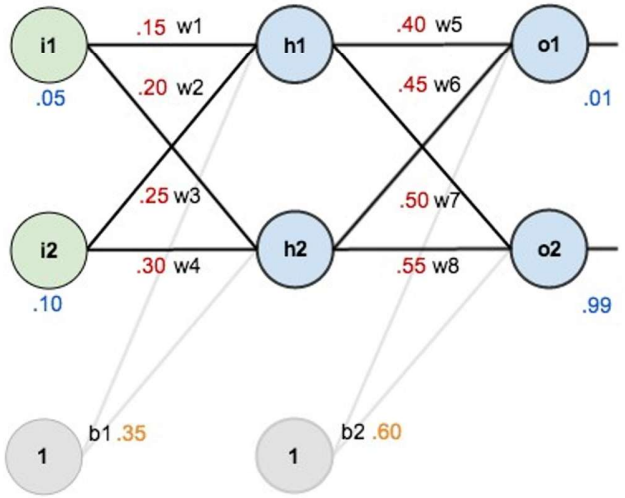


**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Back propagation Assignment

- The goal of backpropagation is to optimize the weights so that the neural network can learn how to correctly map arbitrary inputs to outputs.

- For the rest of this tutorial, we're going to work with a single training set: given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99 and η=0.1






**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# ANN CODE With Python

```
from math import exp
from random import uniform
import numpy as np

def forwardPass(wij,wjk,wkl):
    while(True):
        hiIn = additionBious(np.matmul(input, wij), m)# Hidden Layer 1
        hi = relu(hiIn)
        hjIn = additionBious(np.matmul(hi, wjk), o) # Hidden Layer 2
        hj = sigmoid(hjIn)
        hkIn =additionBious(np.matmul(hj,wkl),p) # Output Layer
        hk = softMax(hkIn)
        if float(format(hk[0],"0.2"))!= y[0] or  float(format(hk[1],"0.2"))!= y[1]:
            print (format(hk[0], "0.4"))+" : "+(format(hk[1],"0.4"))
            wij, wjk, wkl=backwardPass(wij,wjk,wkl,hiIn,hjIn,hkIn,hi,hj,hk)
            #print wjk
        else:
            print (format(hk[0], "0.2"))+(format(hk[1],"0.2"))
            break

def backwardPass(wij,wjk,wkl,hiIn,hjIn,hkIn,hi,hj,hk):
    nwkl,oOut,oIn = outputLayerToHiddenlayer2(wkl, hkIn, hk,hj)
    nwjk,h2In,eTotal=hiddenLayer2ToHiddenLayer1(wjk,wkl,hjIn,oOut,oIn,hi)
    nwij, e2Total=hiddenLayer1ToInput(wij, wjk, eTotal, h2In, hi)
    #print nwjk
    return nwij,nwjk,nwkl


def hiddenLayer1ToInput(wij,wjk,eTotal,h2In,hi):
    e2Total = []
    nwij = np.zeros((n, m), dtype='float')
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
    for i in range(m):
        e2Total.append(0)
        for j in range(o):
            e2Total[i] = e2Total[i] + (eTotal[i] * h2In[i] * wjk[i][j])
    for i in range(m):
        for j in range(o):
            nwij[i][j] = wij[i][j] - (learningRate * (e2Total[j] * hi[j] * input[i]))
    return nwij,e2Total

def hiddenLayer2ToHiddenLayer1(wjk,wkl,hjIn,oOut,oIn,hi):
    h2In=[]
    eTotal=[]
    nwjk = np.zeros((m, o), dtype='float')
    for i in range(o):
        h2In.append(sigmoid1(hjIn[i])*(1-sigmoid1(hjIn[i])))

    for i in range(o):
        eTotal.append(0)
        for j in range(p):
            eTotal[i]=eTotal[i]+(oOut[j] * oIn[j] * wkl[i][j])
    for i in range(m):
        for j in range(o):
            nwjk[i][j]=wjk[i][j]-(learningRate* (eTotal[j]*h2In[j]*hi[i]))
    return nwjk,h2In,eTotal


def outputLayerToHiddenlayer2(wkl,hkIn,hk,hj):
    oOut = []
    oIn = []
    nwkl = np.zeros((o, p), dtype='float')
    for i in range(p):
        oOut.append(-1 * ((y[i] * (1 / hk[i])) + ((1 - y[i]) * (1 / (1 - hk[i])))))
        oIn.append((exp(hkIn[0]) * exp(hkIn[1])) / ((exp(hkIn[0]) + exp(hkIn[1])) * (exp(hkIn[0]) +
exp(hkIn[1]))))
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
    for i in range(o):
        for j in range(p):
            nwkl[i][j] =wkl[i][j]-(learningRate* (oOut[j] * oIn[j] * hj[j]))
    return nwkl,oOut,oIn
```

```
def relu(hi):
    for i in range(m):
        hi[i]=max(hi[i],0)
    return hi
def sigmoid(hj):
    for i in range(o):
        hj[i]=1/(1+exp(-hj[i]))
    return hj
def sigmoid1(w):
    w=1/(1+exp(-w))
    return w
def softMax(hk):
    sum=0
    for i in range(p):
        sum=sum+exp(hk[i])
    try:
        for i in range(p):
            hk[i] =exp(hk[i])/sum
    except:
        return hk
    return hk
def additionBious(w,n):
    for i in range(n):
        w[i]=w[i]+bious
    return w
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
def rnadomWightGenerator():
    wij = np.zeros((n, m), dtype='float')
    wjk = np.zeros((m, o), dtype='float')
    wkl = np.zeros((o,p), dtype='float')
    for i in range(n):
        for j in range(m):
            wij[i][j]=(float(format(uniform(.1, .9), '.3f')))
    for i in range(m):
        for j in range(o):
            wjk[i][j]=(float(format(uniform(.1, .9), '.3f')))
    for i in range(o):
        for j in range(p):
            wkl[i][j]=(float(format(uniform(.1, .9), '.3f')))
    forwardPass(wij,wjk,wkl)

n=13
m=3
o=3
p=2
bious=1
learningRate=-0.01
input=[.1,.2,.7,.8,.45,-.64,.32,.1,.2,-.7,.8,-.45,-.64]
y=[0.0,1.0]
rnadomWightGenerator()
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# WEEK-12
# Fuzzy logic control in an automated car system



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Fuzzification

*Parameters of the fuzzy sets*

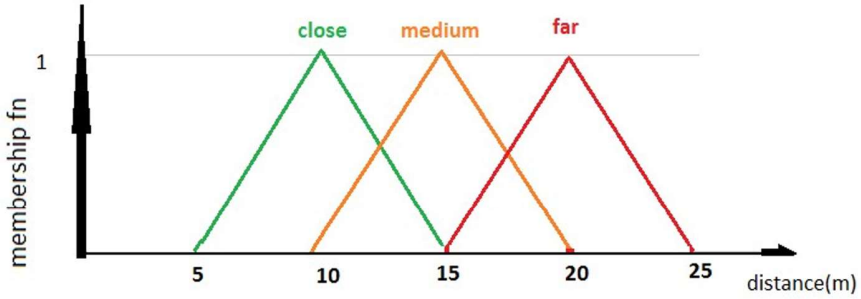
*SPEED:*
*slow [20 to 60 kmph]*
*Medium [40 to 80 kmph]*
*fast [60 to 100 kmph]*

*OBSTACLE DISTANCE:*
*Close [5 to 15 meters]*
*Medium [10 to 20 meters]*
*Far [15 to 25 meters]*

• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha


• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

# Rules

Rule 1: If (obstacle is far)&(speed is slow) then angle is straight

Rule 2: If (obstacle is far)&(speed is medium) then angle is smallT

Rule 3: If (obstacle is far)&(speed is fast) then angle is medT

Rule 4: If (obstacle is med)&(speed is slow) then angle is smallT

Rule 5: If (obstacle is med)&(speed is medium) then angle is medT

Rule 6: If (obstacle is med)&(speed is fast) then angle is bigT

Rule 7: If (obstacle is close)&(speed is slow) then angle is medT

Rule 8: If (obstacle is close)&(speed is medium) then angle is bigT

Rule 9: If (obstacle is close)&(speed is fast) then angle is bigT

• Md. Riadul Islam Assistant Professor, Dept. of CSE, UGV



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```python
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl

# Define fuzzy input variables
speed = ctrl.Antecedent(np.arange(0, 121, 1), 'speed')  # km/h
distance = ctrl.Antecedent(np.arange(0, 51, 1), 'distance')  # meters
brake = ctrl.Consequent(np.arange(0, 101, 1), 'brake')  # percentage

# Define membership functions for speed
speed['slow'] = fuzz.trimf(speed.universe, [0, 0, 40])
speed['medium'] = fuzz.trimf(speed.universe, [30, 60, 90])
speed['fast'] = fuzz.trimf(speed.universe, [80, 120, 120])

# Define membership functions for distance
distance['near'] = fuzz.trimf(distance.universe, [0, 0, 15])
distance['medium'] = fuzz.trimf(distance.unive ↓  [10, 25, 40])
distance['far'] = fuzz.trimf(distance.universe, [30, 50, 50])
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
# Define membership functions for brake intensity
brake['low'] = fuzz.trimf(brake.universe, [0, 0, 40])
brake['medium'] = fuzz.trimf(brake.universe, [30, 50, 70])
brake['high'] = fuzz.trimf(brake.universe, [60, 100, 100])

# Define fuzzy rules
rule1 = ctrl.Rule(speed['slow'] & distance['far'], brake['low'])
rule2 = ctrl.Rule(speed['slow'] & distance['medium'], brake['medium'])
rule3 = ctrl.Rule(speed['slow'] & distance['near'], brake['high'])

rule4 = ctrl.Rule(speed['medium'] & distance['far'], brake['low'])
rule5 = ctrl.Rule(speed['medium'] & distance['medium'], brake['medium'])
rule6 = ctrl.Rule(speed['medium'] & distance['near'], brake['high'])

rule7 = ctrl.Rule(speed['fast'] & distance['far'], brake['medium'])
rule8 = ctrl.Rule(speed['fast'] & distance['medium'], brake['high'])
rule9 = ctrl.Rule(speed['fast'] & distance['near'], brake['high'])
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha

```
# Build control system
braking_ctrl = ctrl.ControlSystem([rule1, rule2, rule3,
                                   rule4, rule5, rule6,
                                   rule7, rule8, rule9])
braking = ctrl.ControlSystemSimulation(braking_ctrl)

# Input values
braking.input['speed'] = 45  # km/h
braking.input['distance'] = 12  # meters

# Compute result
braking.compute()

print(f"Brake intensity: {braking.output['brake']:.2f}%")
brake.view(sim=braking)
```



**Prepared By:** Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barisha