AI lab manual 2019 pattern

Computer Engineering (Savitribai Phule Pune University)

Scan to open on Studocu

Studocu is not sponsored or endorsed by any college or university
Downloaded by Shweta Amale (shweta.amale@dypic.in)

A

LABORATORY MANUAL

On

"LABORATORY PRACTICE II (310258)"

TE SEMESTER – (VI)

Prepared by: Ms. P. P. Boraste

Checked by: Dr. D. V. Medhane

NAME OF LABORATORY: LABORATORY PRACTICE II (310258)

DEPARTMENT OF COMPUTER ENGINEERING

Nashik District Maratha VidyaPrasarakSamaj's

Karmaveer Adv. BaburaoGanpatraoThakare College of Engineering

Nashik

A.Y. 2021-22

MVPS's KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Vision and Mission

Vision of Computer Department

To be the center for excellence for training the world-class engineers to work with multidisciplinary domain based on the state-of-the-art of technology enabled academic system blended with industrial and business practices.

Mission of Computer Department

To educate and train undergraduate students in Computer Engineering by instilling excellence to fulfill professional and social requirements in business and industry on the platform of scientifically designed academic processes.

Program Educational Objectives

1. To inculcate computational and programming skills in the field of Computer Engineering..
2. To prepare the graduates to fulfill professional requirements in industry.
3. To develop the graduates to solve problems related to the society

MVPS

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Program Outcomes

PO1	Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

MVPS KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Program Specific Outcomes

PSO1	To apply mathematical and Computer Engineering fundamentals.
PSO2	To apply standard practices and strategies for software development and project management
PSO3	To adapt programming languages, modern computer tools and technologies and soft skills for career enrichment.

Course Outcomes:

CO1	Design a system using different informed search / uninformed search or heuristic approaches
CO2	Apply basic principles of AI in solutions that require problem solving, inference, perception, knowledge representation, and learning.
CO3	Design and develop an interactive AI application

CO and Assignment mapping

Course outcomes	**After successful Completion of the course, student will be able to**	**Bloom's Taxonomy Level**	**Experiments Mapped**	**Target Set**
CO1	Design a system using different informed search / uninformed search or heuristic approaches	Apply(3)		
CO2	Apply basic principles of AI in solutions that require problem solving, inference, perception, knowledge representation, and learning.	Apply(3)		
CO3	Design and develop an interactive AI application	Apply(3)		

MVPS'SKBTCOE

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Rubrics for term work

Sr. No.	Criteria	Good	Average	Below Average	Marks (10)
1	Timely completion, punctuality	Submitted within deadline (2)	Late submission (1)	Late submission (1)	2
2	Performance, involvement, efficiency	Followed proper steps accurately with indentation and neat formatting, sufficient use of language features, Logical thinking, Overall understanding (5-6)	Followed steps partially, but with less indentation / formatting, , Logical thinking, Overall understanding (3-4)	Not followed proper steps, no indentation and formatting, no comments, Logical thinking, Overall understanding (1-2)	6
3	Documentation, neatness	Proper documentation and neatness followed (2)	Documentation not proper or neatness not observed (1)	Documentation not proper or neatness not observed (1)	2

MVPS'S KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Nashik District Maratha VidyaPrasarakSamaj's

KARMAVEER ADV. BABURAO GANPATRAO THAKARE

COLLEGE OF ENGINEERING, NASHIK

Udoji Maratha Boarding Campus, Near Pumping station, Gangapur Road

DEPARTMENT OF COMPUTER ENGINEERING

SUBJECT: LABORATORY PRACTICE II (310258)

INDEX

MVPS KBTCOE

Part 1:Artificial Intelligence
Group A(All assignments are compulsory)

Sr. No.	TITLE	PAGE NO.	DATE	R1	R2	R3	TOTAL	SIGN
1	Implement depth first search algorithm and Breadth First Search algorithm, Use an undirected graph and develop a recursive algorithm for searching all the vertices of a graph or tree data structure.							
2	Implement A star Algorithm for any game search problem.							
3	Implement Greedy search algorithm for any of the following application: 1. Selection Sort 2. Minimum Spanning Tree 3. Single-Source Shortest Path Problem 4. Job Scheduling Problem 5. Prim's Minimal Spanning Tree Algorithm 6. Kruskal's Minimal Spanning Tree Algorithm 7. Dijkstra's Minimal Spanning Tree Algorithm							
Group B								
4	Implement a solution for a Constraint Satisfaction Problem							

	using Branch and Bound and Backtracking for n-queens problem or a graph coloring problem.							
5	Develop an elementary catboat for any suitable customer interaction application.							
Group C								
6	Implement any one of the following Expert System 1. Information management 2. Hospitals and medical facilities 3. Help desks management 4. Employee performance evaluation 5. Stock market trading 6. Airline scheduling and cargo schedules							
	Rubrics: R1Timelycompletion, punctuality R2 -Performance, innovation, efficiency R3- Documentation, neatness	**Average Marks**						

MVP'S KBTCOE

Certificate

This is to certify that Mr/Ms.____________________________________Roll No.:________, students of TE Computer Engineering, has completed the above said experiments/Term work for semester-II of the academic year 2021-22.

PRN NO:	EXAMINATION NO:

Ms.P. P. Boraste
Subject Incharge

Dr. D. V. Medhane
Head of Dept.

Dr. S. R. Devane
Principal

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Experiment No: 01

1.1 Aim:

Implement depth first search algorithm and Breadth First Search algorithm, Use an undirected Graph and develop a recursive algorithm for searching all the vertices of a graph or tree data structure.

1.2 Objectives:

To implement depth first search algorithm and Breadth First Search algorithm and develop a recursive algorithm

1.3 Software used (if applicable) / Programming Languages Used: 64-bit Windows OS and Linux , C++/,Java, python

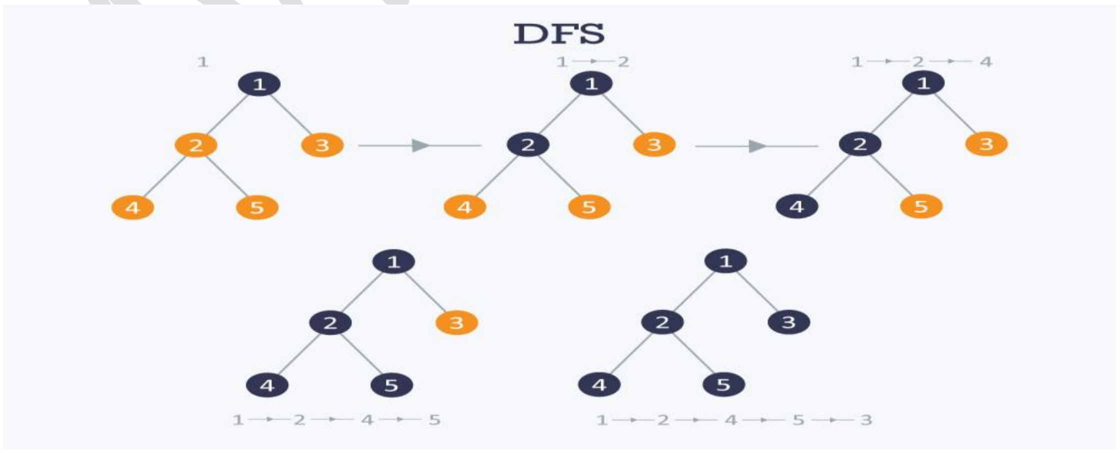
1.4. Theory:

Depth First Search (DFS) Algorithm

The DFS algorithm is a recursive algorithm that uses the idea of backtracking. It involves exhaustive searches of all the nodes by going ahead, if possible, else by backtracking.

Here, the word backtrack means that when you are moving forward and there are no more nodes along the current path, you move backwards on the same path to find nodes to traverse. All the nodes will be visited on the current path till all the unvisited nodes have been traversed after which the next path will be selected.

This recursive nature of DFS can be implemented using stacks. The basic idea is as follows: Pick a starting node and push all its adjacent nodes into a stack. Pop a node from stack to select the next node to visit and push all its adjacent nodes into a stack. Repeat this process until the stack is empty. However, ensure that the nodes that are visited are marked. This will prevent you from visiting the same node more than once. If you do not mark the nodes that are visited and you visit the same node more than once, you may end up in an infinite loop.



Downloaded by Shweta Amale (shweta.amale@dypic.in)

Time complexity O(V+E), when implemented using an adjacency list.

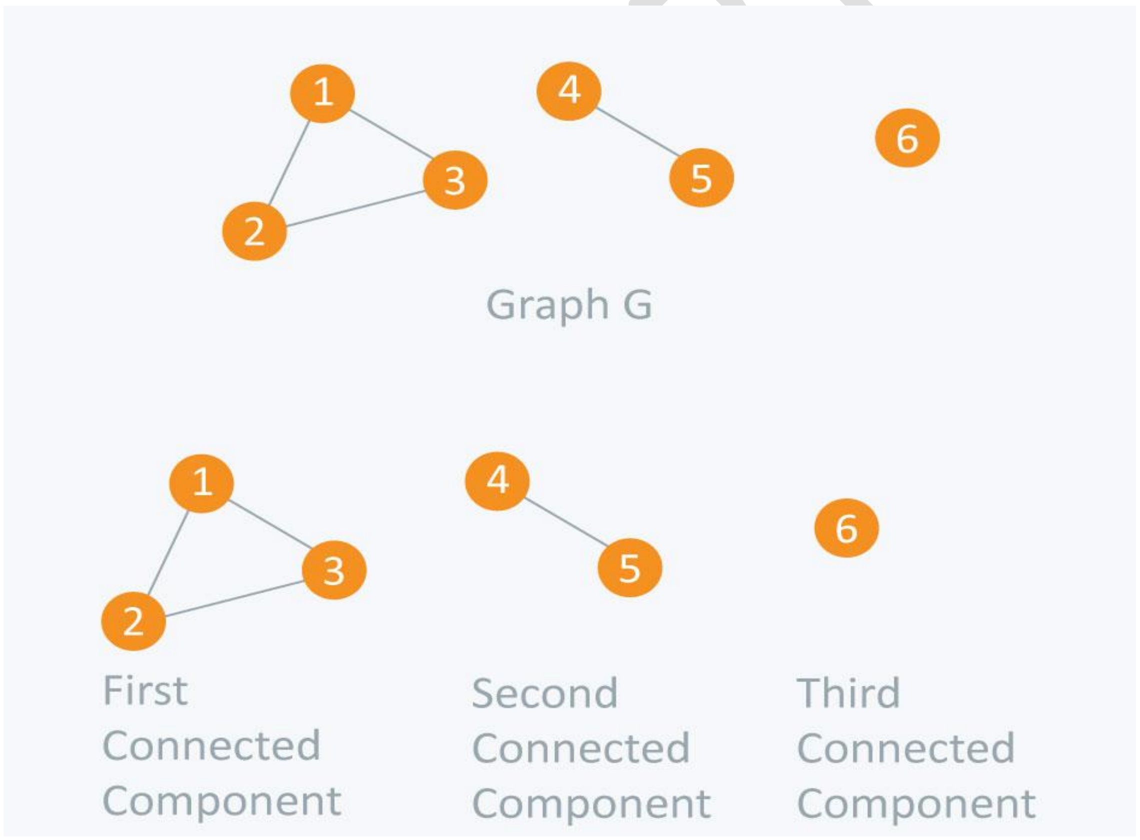
How to find connected components using DFS?

A graph is said to be disconnected if it is not connected, i.e. if two nodes exist in the graph such that there is no edge in between those nodes. In an undirected graph, a connected component is a set of vertices in a graph that are linked to each other by paths.

Consider the example given in the diagram. Graph G is a disconnected graph and has the following 3 connected components.

- First connected component is 1 -> 2 -> 3 as they are linked to each other
- Second connected component 4 -> 5
- Third connected component is vertex 6

In DFS, if we start from a start node it will mark all the nodes connected to the start node as visited. Therefore, if we choose any node in a connected component and run DFS on that node it will mark the whole connected component as visited.


1
2
3
4
5
6
Graph G
1
2
3
4
5
6
First Connected Component
Second Connected Component
Third Connected Component

Depth first search (DFS) algorithm starts with the initial node of the graph G, and then goes to deeper and deeper until we find the goal node or the node which has no children. The algorithm, then backtracks from the dead end towards the most recent node that is yet to be completely unexplored.

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

The data structure which is being used in DFS is stack. The process is similar to BFS algorithm. In DFS, the edges that lead to an unvisited node are called discovery edges while the edge that leads to an already visited node are called block edges.

DFS Algorithm

The recursive method of the Depth-First Search algorithm is implemented using stack. A standard Depth-First Search implementation puts every vertex of the graph into one in all 2 categories: 1) Visited 2) Not Visited. **The only purpose of this algorithm is to visit all the vertex of the graph avoiding cycles.**

The DSF algorithm follows as:

1. We will start by putting any one of the graph's vertex on top of the stack.
2. After that take the top item of the stack and add it to the visited list of the vertex.
3. Next, create a list of that adjacent node of the vertex. Add the ones which aren't in the visited list of vertexes to the top of the stack.

Lastly, keep repeating steps 2 and 3 until the stack is empty

- **Step 1:** SET STATUS = 1 (ready state) for each node in G
- **Step 2:** Push the starting node A on the stack and set its STATUS = 2 (waiting state)
- **Step 3:** Repeat Steps 4 and 5 until STACK is empty
- **Step 4:** Pop the top node N. Process it and set its STATUS = 3 (processed state)
- **Step 5:** Push on the stack all the neighbours of N that are in the ready state (whose
  STATUS = 1) and set their
  STATUS = 2 (waiting state)
  [END OF LOOP]
- **Step 6:** EXIT

Time Complexity

The time complexity of the Depth-First Search algorithm is represented within the sort of **O(V + E)**, where V is that the number of nodes and E is that the number of edges.

The space complexity of the algorithm is **O(V).**

Applications

Depth-First Search Algorithm has a wide range of applications for practical purposes. Some of them are as discussed below:

1. For finding the strongly connected components of the graph
2. For finding the path

Downloaded by Shweta Amale (shweta.amale@dypic.in)

3. To test if the graph is bipartite
4. For detecting cycles in a graph
5. Topological Sorting
6. Solving the puzzle with only one solution.
7. Network Analysis
8. Mapping Routes
9. Scheduling a problem

Breadth First Search (BFS) Algorithm

Graph traversals

Graph traversal means visiting every vertex and edge exactly once in a well-defined order. While using certain graph algorithms, you must ensure that each vertex of the graph is visited exactly once. The orders in which the vertices are visited are important and may depend upon the algorithm or question that you are solving.

During a traversal, it is important that you track which vertices have been visited. The most common way of tracking vertices is to mark them.

Breadth First Search (BFS)

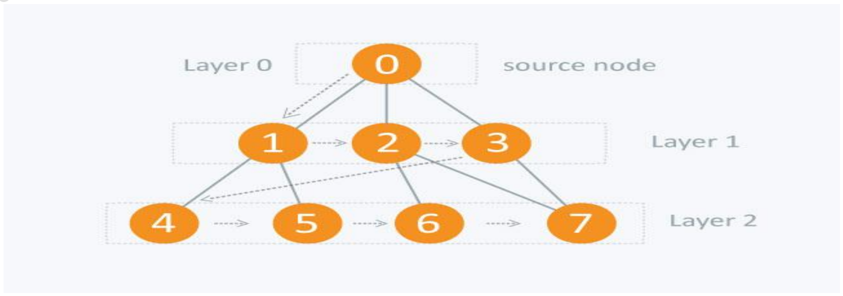
Breadth-First Search (BFS) is an algorithm used for traversing graphs or trees. Traversing means visiting each node of the graph, Breadth-First Search is a recursive algorithm to search all the vertices of a graph or a tree. **BFS in python can be implemented by using data structures like a dictionary and lists.** Breadth-First Search in tree and graph is almost the same. The only difference is that the graph may contain cycles, so we may traverse to the same node again

There are many ways to traverse graphs. BFS is the most commonly used approach.

BFS is a traversing algorithm where you should start traversing from a selected node (source or starting node) and traverse the graph layer wise thus exploring the neighbor nodes (nodes which are directly connected to source node). You must then move towards the next-level neighbor nodes.

As the name BFS suggests, you are required to traverse the graph breadthwise as follows:

1. First move horizontally and visit all the nodes of the current layer
2. Move to the next layer Consider the following diagram.



MVPS KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

The distance between the nodes in layer 1 is comparatively lesser than the distance between the nodes in layer 2. Therefore, in BFS, you must traverse all the nodes in layer 1 before you move to the nodes in layer 2.

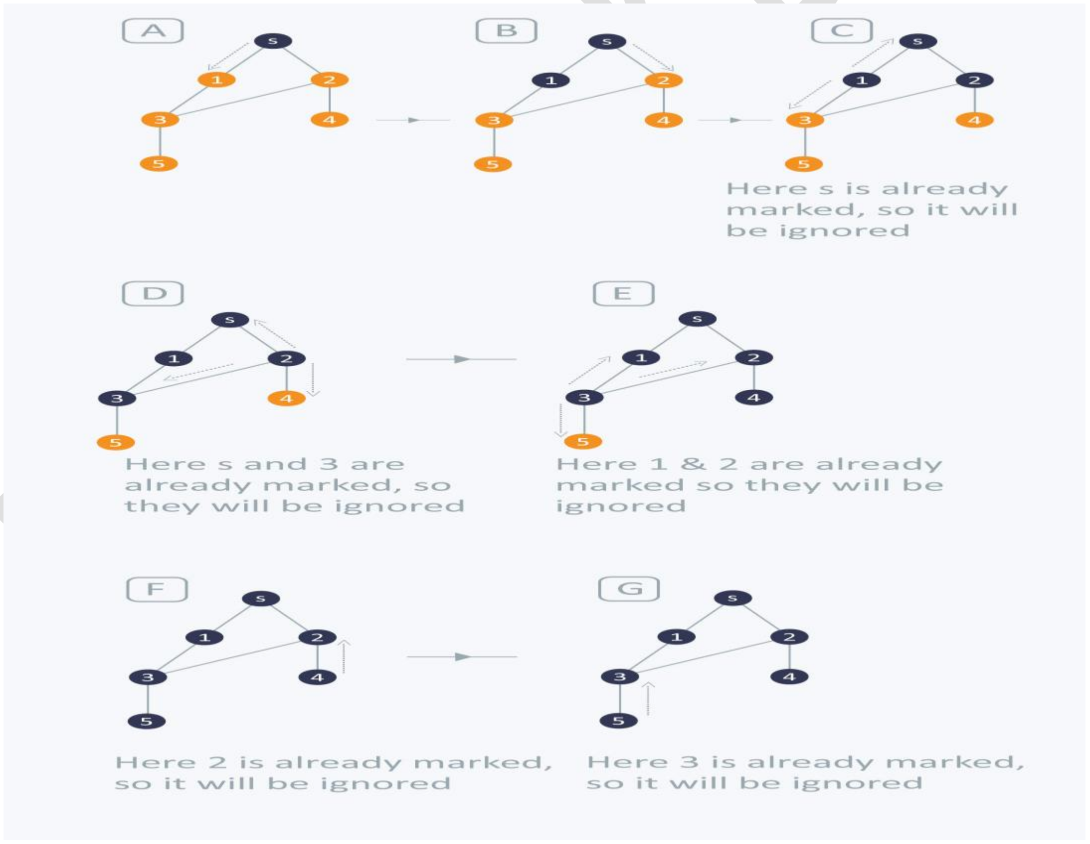
Traversing child nodes

A graph can contain cycles, which may bring you to the same node again while traversing the graph. To avoid processing of same node again, use a Boolean array which marks the node after it is processed. While visiting the nodes in the layer of a graph, store them in a manner such that you can traverse the corresponding child nodes in a similar order.

In the earlier diagram, start traversing from 0 and visit its child nodes 1, 2, and 3, Store them in the order in which they are visited. This will allow you to visit the child nodes of 1 first (i.e. 4 and 5), then of 2 (i.e. 6 and 7), and then of 3 (i.e. 7) etc.

To make this process easy, use a queue to store the node and mark it as 'visited' until its entire neighbor's (vertices that are directly connected to it) are marked. The queue follows the First in First out (FIFO) queuing method, and therefore, the neighbors of the node will be visited in the order in which they were inserted in the node i.e. the node that was inserted first will be visited first, and so on.

Traversing Process


A
B
C
D
E
F
G
s
1
2
3
4
5
Here s is already marked, so it will be ignored
Here s and 3 are already marked, so they will be ignored
Here 1 & 2 are already marked so they will be ignored
Here 2 is already marked, so it will be ignored
Here 3 is already marked, so it will be ignored

Downloaded by Shweta Amale (shweta.amale@dypic.in)

The traversing will start from the source node and push *s* in queue. *s* will be marked as 'visited'.

First iteration

- s will be popped from the queue
- Neighbors of s i.e. 1 and 2 will be traversed
- 1 and 2, which have not been traversed earlier, are traversed. They will be:
    - Pushed in the queue
    - 1 and 2 will be marked as visited

Second iteration

- 1 is popped from the queue
- Neighbors of 1 i.e. s and 3 are traversed
- s is ignored because it is marked as 'visited'
- 3, which have not been traversed earlier, is traversed. It is:
    - Pushed in the queue
    - Marked as visited

Third iteration

- 2 is popped from the queue
- Neighbors of 2 i.e. s, 3, and 4 are traversed
- 3 and s are ignored because they are marked as 'visited'
- 4, which have not been traversed earlier, is traversed. It is:
    - Pushed in the queue
    - Marked as visited

Fourth iteration

- 3 is popped from the queue
- Neighbors of 3 i.e. 1, 2, and 5 are traversed
- 1 and 2 are ignored because they are marked as 'visited'
- 5, which have not been traversed earlier, is traversed. It is:
    - Pushed in the queue
    - Marked as visited

Fifth iteration

- 4 will be popped from the queue
- Neighbors of 4 i.e. 2 is traversed
- 2 is ignored because it is already marked as 'visited'

Sixth iteration

- 5 is popped from the queue

MPSC KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

- Neighbors of 5 i.e. 3 is traversed
- 3 is ignored because it is already marked as 'visited'

The queue is empty and it comes out of the loop. All the nodes have been traversed by using BFS.

If all the edges in a graph are of the same weight, then BFS can also be used to find the minimum distance between the nodes in a graph.

Time Complexity

The time complexity of the Breadth first Search algorithm is in the form of **O(V+E)**, where V is the representation of the number of nodes and E is the number of edges.

Also, the space complexity of the BFS algorithm is **O(V).**

Applications

Breadth-first Search Algorithm has a wide range of applications in the real-world. Some of them are as discussed below:

1. In GPS navigation, it helps in finding the shortest path available from one point to another.
2. In path finding algorithms
3. Cycle detection in an undirected graph
4. In minimum spanning tree
5. To build index by search index
6. In Ford-Fulkerson algorithm to find maximum flow in a network.

Conclusion

Hence, we have Implemented depth first search algorithm and Breadth First Search algorithm, using an undirected graph and develop a recursive algorithm for searching all the vertices of a graph or tree data structure.

Staff signature with date

Question
1) What is Depth First Search? Explain it with example?
2) What is Breadth First Search? Explain it with example?
3) Describe concept of backtracking in detail?

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Experiment No: 02

1.1 Aim:
Implement A star Algorithm for any game search problem.

1.2 Objectives:
Implement A star Algorithm for any game search problem.

1.3 Software used (if applicable) / Programming Languages Used: 64-bit Windows OS and Linux , C++/,Java, python

1.4. Theory:

What is the A* algorithm?

A * algorithm is a searching algorithm that searches for the shortest path between the *initial and the final state.* It is used in various applications, such as *maps*.

In *maps* the A* algorithm is used to calculate the shortest distance between the source (initial state) and the destination (final state).

The most important advantage of A* search algorithm which separates it from other traversal techniques is that it has a brain. This makes A* very smart and pushes it much ahead of other conventional algorithms.

How it works

Imagine a square grid which possesses many obstacles, scattered randomly. The initial and the final cell are provided. The aim is to reach the final cell in the shortest amount of time.

Here A* Search Algorithm comes to the rescue:

Explanation

A* algorithm has 3 parameters:

- **g:** the cost of moving from the initial cell to the current cell. Basically, it is the sum of all the cells that have been visited since leaving the first cell.
- **h:** also known as the *heuristic value,* it is the **estimated** cost of moving from the current cell to the final cell. The actual cost cannot be calculated until the final cell is reached. Hence, h is the estimated cost. We **must** make sure that there is **never** an over estimation of the cost.
- **f:** it is the sum of g and h. So, **f = g + h**

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

The way that the algorithm makes its decisions is by taking the f-value into account. The algorithm selects the *smallest f-valued cell* and moves to that cell. This process continues until the algorithm reaches its goal cell.

A* Algorithm works by vertices in the graph which start with the starting point of the object and then repeatedly examines the next unexamined vertex, adding its vertices to the set of vertices that will be examined. A* Algorithm is popular because it is a technique that is used for finding path and graph traversals. This algorithm is used by many web-based maps and games.

A* Algorithm Steps

Step1: Place the starting node in the OPEN list.
Step 2: Check if the OPEN list is empty or not, if the list is empty then return failure and stops.
Step 3: Select the node from the OPEN list which has the smallest value of evaluation function (g+h), if node n is goal node then return success and stop, otherwise
Step 4: Expand node n and generate all of its successors, and put n into the closed list. For each successor n', check whether n' is already in the OPEN or CLOSED list, if not then compute evaluation function for n' and place into Open list.
Step 5: Else if node n' is already in OPEN and CLOSED, then it should be attached to the back pointer which reflects the lowest g(n') value.
Step 6: Return to Step 2.

Advantages:
1. A* search algorithm is the best algorithm than other search algorithms.
2. A* search algorithm is optimal and complete.
3. This algorithm can solve very complex problems.

Disadvantages:
1. It does not always produce the shortest path as it mostly based on heuristics and approximation.
2. A* search algorithm has some complexity issues.
3. The main drawback of A* is memory requirement as it keeps all generated nodes in the memory, so it is not practical for various large-scale problems.

Conclusion
Hence, we have implemented A* Algorithm for any game search problem.

Staff signature with date

Question
1) What is A* Algorithm? How does it work?
2) Why is A* algorithm popular?
3) What is a Heuristic Function?

MVPS KBTCOE

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Experiment No: 03

1.1 Aim:

Implement Greedy search algorithm for any of the following application:

I. Selection Sort
II. Minimum Spanning Tree
III. Single-Source Shortest Path Problem
IV. Job Scheduling Problem
V. Prim's Minimal Spanning Tree Algorithm
VI. Kruskal's Minimal Spanning Tree Algorithm
VII. Dijkstra's Minimal Spanning Tree Algorithm

1.2 Objectives:

Implement Greedy search algorithm for given application.

1.3 Software used (if applicable) / Programming Languages Used: 64-bit Windows OS and Linux, C++/, Java, python

1.4. Theory:

What Is Greedy Algorithm?

Greedy algorithm is a problem-solving strategy that makes locally optimal decisions at each stage in the hopes of achieving a globally optimum solution. This simple, intuitive algorithm can be applied to solve any optimization problem which requires the maximum or minimum optimum result. The best thing about this algorithm is that it is easy to understand and implement.

The runtime complexity associated with a greedy solution is pretty reasonable. However, you can implement a greedy solution only if the problem statement follows two properties mentioned below:

- Greedy Choice Property: Choosing the best option at each phase can lead to a global (overall) optimal solution.
- Optimal Substructure: If an optimal solution to the complete problem contains the optimal solutions to the sub problems, the problem has an optimal substructure.

Moving forward, we will learn how to create a greedy solution for a problem that adheres to the principles listed above.

Steps for Creating a Greedy Algorithm

By following the steps given below, you will be able to formulate a greedy solution for the given problem statement:

- Step 1: In a given problem, find the best substructure or sub problem.
- Step 2: Determine what the solution will include (e.g., largest sum, shortest path).

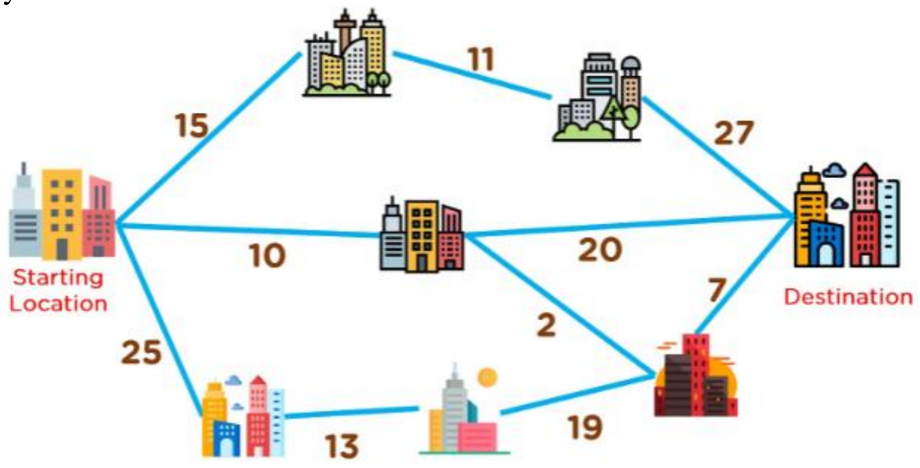
This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

- Step 3: Create an iterative process for going over all sub problems and creating an optimum solution.

Example of Greedy Algorithm

Problem Statement: Find the best route to reach the destination city from the given starting point using a greedy method.

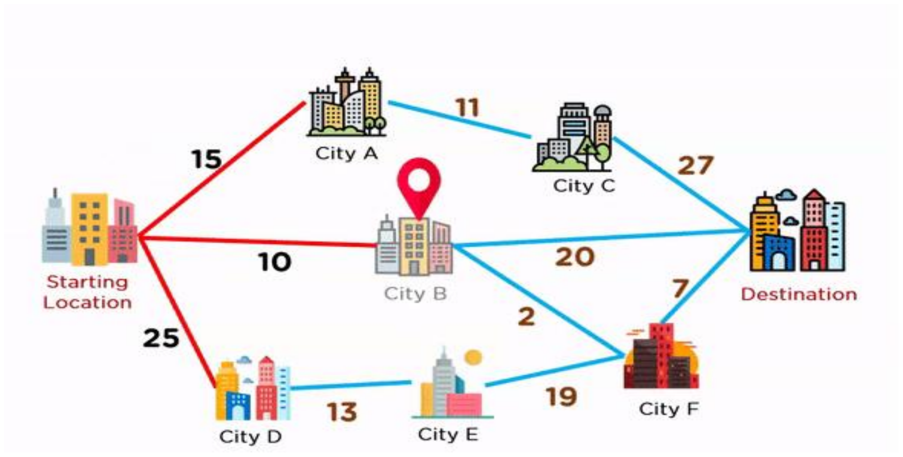


Greedy Solution: In order to tackle this problem, we need to maintain a graph structure. And for that graph structure, we'll have to create a tree structure, which will serve as the answer to this problem. The steps to generate this solution are given below:

- Start from the source vertex.
- Pick one vertex at a time with a minimum edge weight (distance) from the source vertex.
- Add the selected vertex to a tree structure if the connecting edge does not form a cycle.
- Keep adding adjacent fringe vertices to the tree until you reach the destination vertex.

The animation given below explains how paths will be picked up in order to reach the destination city.

Downloaded by Shweta Amale (shweta.amale@dypic.in)



Applications of Greedy Algorithm

Following are few applications of the greedy algorithm:

- Used for Constructing Minimum Spanning Trees: Prim's and Kruskal's Algorithms used to construct minimum spanning trees are greedy algorithms.
- Used to Implement Huffman Encoding: A greedy algorithm is utilized to build a Huffman tree that compresses a given image, spreadsheet, or video into a lossless compressed file.
- Used to Solve Optimization Problems: Graph - Map Coloring, Graph - Vertex Cover, Knapsack Problem, Job Scheduling Problem, and activity selection problem are classic optimization problems solved using a greedy algorithmic paradigm.

1. Selection Sort

- Selection sort is one of the easiest approaches to sorting.
- It is inspired from the way in which we sort things out in day to day life.
- It is an in-place sorting algorithm because it uses no auxiliary data structures while sorting.

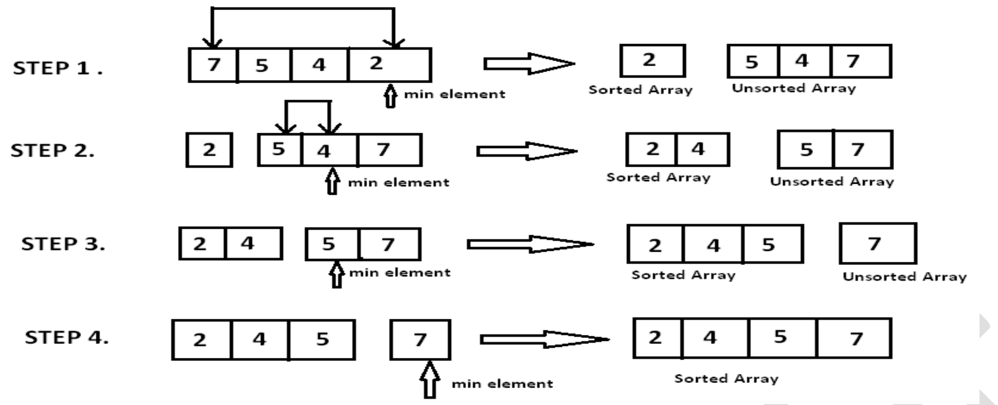
The Selection sort algorithm is based on the idea of finding the minimum or maximum element in an unsorted array and then putting it in its correct position in a sorted array.

Assume that the array A=[7,5,4,2] needs to be sorted in ascending order.

The minimum element in the array i.e. 2 is searched for and then swapped with the element that is currently located at the first position, i.e. 7. Now the minimum element in the remaining unsorted array is searched for and put in the second position, and so on.

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)



II. Minimum Spanning Tree

What is a Spanning Tree?

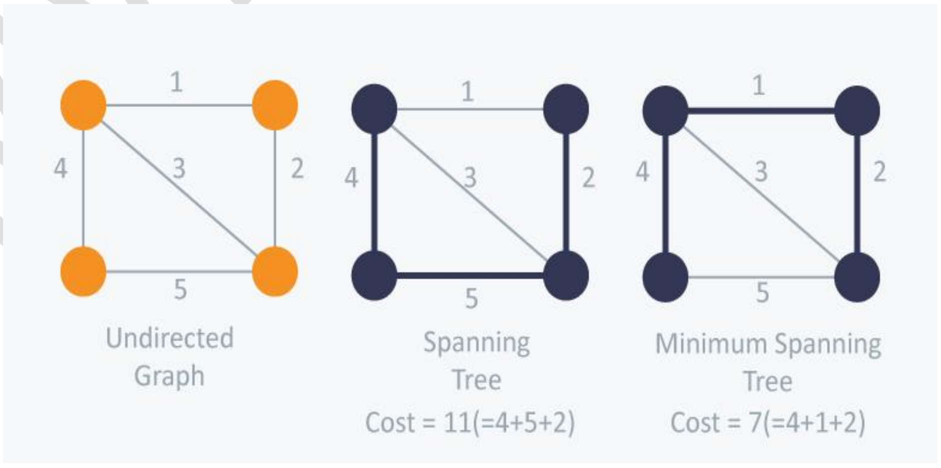
Given an undirected and connected graph G=(V,E), a spanning tree of the graph G is a tree that spans G (that is, it includes every vertex of G) and is a sub graph of G (every edge in the tree belongs to G)

What is a Minimum Spanning Tree?

The cost of the spanning tree is the sum of the weights of all the edges in the tree. There can be many spanning trees. Minimum spanning tree is the spanning tree where the cost is minimum among all the spanning trees. There also can be many minimum spanning trees.

Minimum spanning tree has direct application in the design of networks. It is used in algorithms approximating the travelling salesman problem, multi-terminal minimum cut problem and minimum-cost weighted perfect matching. Other practical applications are:

1. Cluster Analysis
2. Handwriting recognition
3. Image segmentation



Downloaded by Shweta Amale (shweta.amale@dypic.in)

There are two famous algorithms for finding the Minimum Spanning Tree:

III. Single-Source Shortest Path Problem

The Single-Source Shortest Path (SSSP) problem consists of finding the shortest paths between a given vertex *v* and all other vertices in the graph. Algorithms such as Breadth-First-Search (BFS) for unweight graphs or Dijkstra solve this problem. The All-Pairs Shortest Path (APSP) problem consists of finding the shortest path between all pairs of vertices in the graph. To solve this second problem, one can use the Floyd-Warshall algorithm [2] or apply the Dijkstra algorithm to each vertex in the graph.
The Single-Pair Shortest Path (SPSP) problem consists of finding the shortest path between a single pair of vertices. This problem is mostly solved using Dijkstra, though in this case a single result is kept and other shortest paths are discarded.

The single source shortest path algorithm (for arbitrary weight positive or negative) is also known Bellman-Ford algorithm is used to find minimum distance from source vertex to any other vertex. The main difference between these algorithms with Dijkstra's algorithm is, in Dijkstra's algorithm we cannot handle the negative weight, but here we can handle it easily.

IV. Job Scheduling Problem

In job sequencing problem, the objective is to find a sequence of jobs, which is completed within their deadlines and gives maximum profit.

This is the dispute of optimally scheduling unit-time tasks on a single processor, where each job has a deadline and a penalty that necessary be paid if the deadline is missed.

A unit-time task is a job, such as a program to be rush on a computer that needed precisely one unit of time to complete. Given a finite set S of unit-time tasks, a schedule for S is a permutation of S specifying the order in which to perform these tasks. The first task in the schedule starts at time 0 and ends at time 1; the second task begins at time 1 and finishes at time 2, and so on.

The dispute of scheduling unit-time tasks with deadlines and penalties for each processor has the following inputs:

- a set S = {1, 2, 3.....n} of n unit-time tasks.
- a set of n integer deadlines d_1 d_2 d_3...d_n such that d_i satisfies $1 \leq d_i \leq n$ and task i is supposed to finish by time d_i and
- a set of n non-negative weights or penalties w_1 w_2....w_n such that we incur a penalty of w_i if task i is not finished by time d_i, and we incurred no penalty if a task finishes by its deadline.

Here we find a schedule for S that minimizes the total penalty incurred for missed deadlines.

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

A task is **late** in this schedule if it finished after its deadline. Otherwise, the task is early in the schedule. An arbitrary schedule can consistently be put into **early-first form**, in which the first tasks precede the late tasks, i.e., if some new task x follows some late task y, then we can switch the position of x and y without affecting x being early or y being late.

An arbitrary schedule can always be put into a **canonical form** in which first tasks precede the late tasks, and first tasks are scheduled in order of non-decreasing deadlines.

A set A of tasks is **independent** if there exists a schedule for the particular tasks such that no tasks are late. So the set of first tasks for a schedule forms an independent set of tasks 'l' denote the set of all independent set of tasks.

For any set of tasks A, A is independent if for t = 0, 1, 2.....n we have $N_t(A) \leq t$ where $N_t(A)$ denotes the number of tasks in A whose deadline is t or prior, i.e. if the tasks in A are expected in order of monotonically growing deadlines, then no task is late.

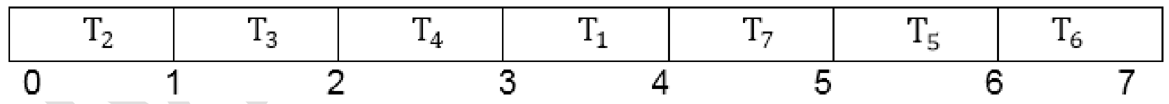
Example: Find the optimal schedule for the following task with given weight (penalties) and deadlines.

	1	**2**	**3**	**4**	**5**	**6**	**7**
d_i	4	2	4	3	1	4	6
w_i	70	60	50	40	30	20	10

Solution: According to the Greedy algorithm we sort the jobs in decreasing order of their penalties so that minimum of penalties will be charged.

In this problem, we can see that the maximum time for which uniprocessor machine will run in 6 units because it is the maximum deadline.

Let T_i represents the tasks where i = 1 to 7



T_5 and T_6 cannot be accepted after T_7 so penalty is

$w_5 + w_6 = 30 + 20 = 50$ (2 3 4 1 7 5 6)

Other schedule is

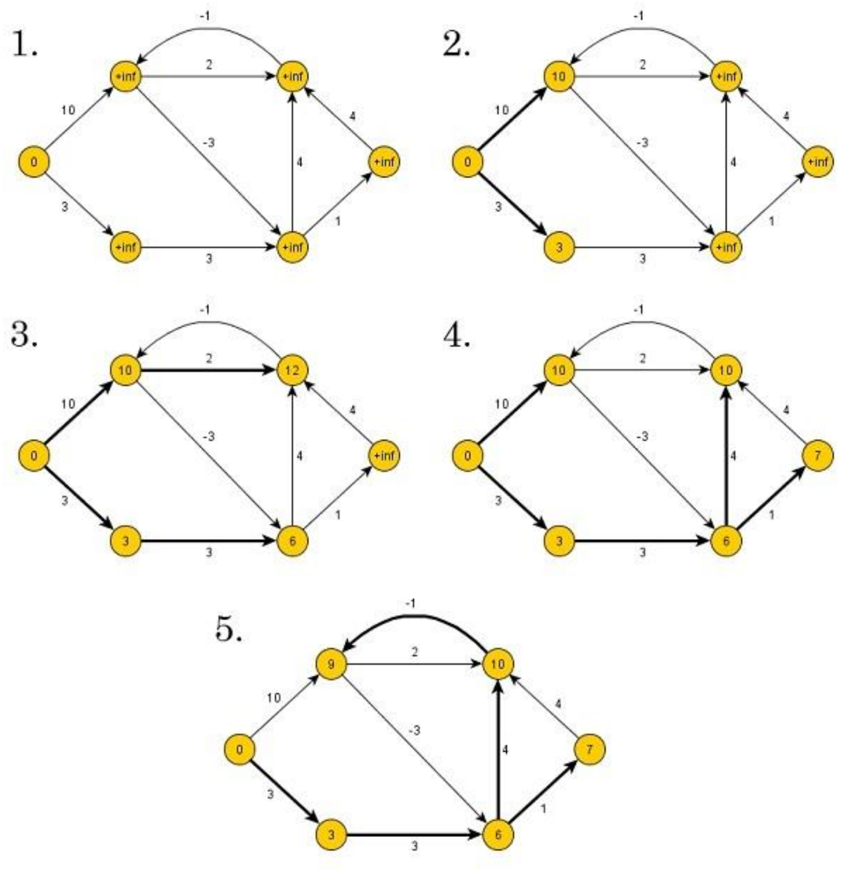
Downloaded by Shweta Amale (shweta.amale@dypic.in)

T_2	T_4	T_1	T_3	T_7	T_5	T_6

0 1 2 3 4 5 6 7

(2 4 1 3 7 5 6)

There can be many other schedules but (2 4 1 3 7 5 6) is optimal.



Bellman-Ford algorithm finds the distance in bottom up manner. At first it finds those distances which have only one edge in the path, after that increase the path length to find all possible solutions.

VI.Prim's Minimal Spanning Tree Algorithm
Prim’s Algorithm

Prim’s Algorithm also uses Greedy approach to find the minimum spanning tree. In Prim’s Algorithm we grow the spanning tree from a starting position. Unlike an **edge** in Kruskal's, we add **vertex** to the growing spanning tree in Prim's.

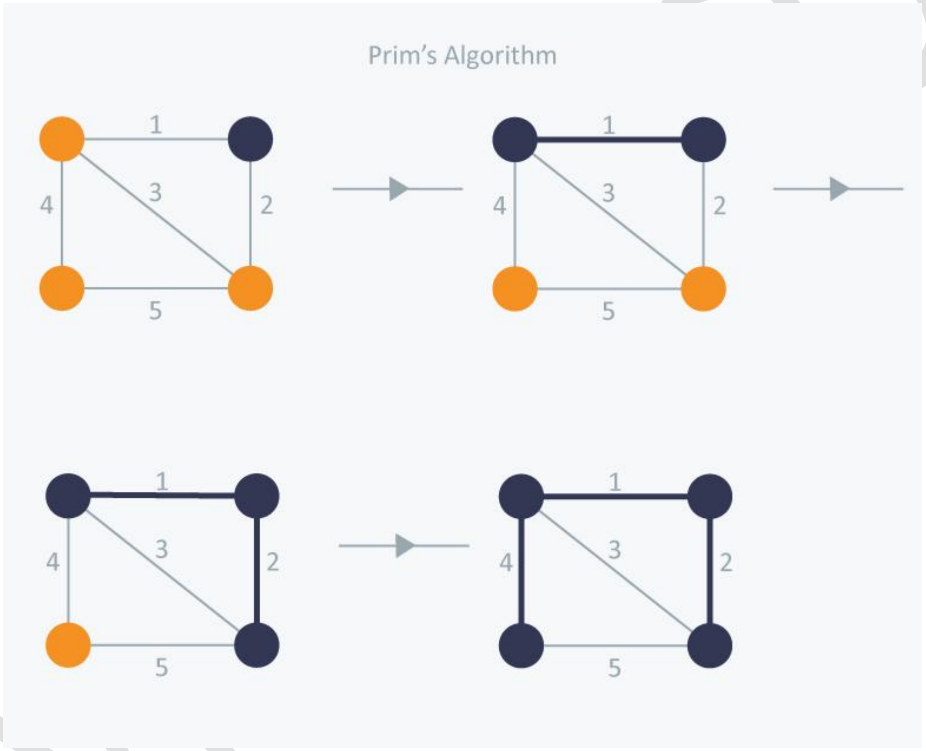
This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Algorithm Steps:

- Maintain two disjoint sets of vertices. One containing vertices that are in the growing spanning tree and other that are not in the growing spanning tree.
- Select the cheapest vertex that is connected to the growing spanning tree and is not in the growing spanning tree and add it into the growing spanning tree. This can be done using Priority Queues. Insert the vertices that are connected to growing spanning tree, into the Priority Queue.
- Check for cycles. To do that, mark the nodes which have been already selected and insert only those nodes in the Priority Queue that are not marked.

Consider the example below:



In Prim's Algorithm, we will start with an arbitrary node (it doesn't matter which one) and mark it. In iteration we will mark a new vertex that is adjacent to the one that we have already marked. As a greedy algorithm, Prim's algorithm will select the cheapest edge and mark the vertex. So we will simply choose the edge with weight 1. In the next iteration we have three options, edges with weight 2, 3 and 4. So, we will select the edge with weight 2 and mark the vertex. Now again we have three options, edges with weight 3, 4 and 5. But we can't choose edge with weight 3 as it is creating a cycle. So we will select the edge with weight 4 and we end up with the minimum spanning tree of total cost 7 (= 1 + 2 +4).

Applications

- Prim's algorithm is used in network design
- It is used in network cycles and rail tracks connecting all the cities
- Prim's algorithm is used in laying cables of electrical wiring

Downloaded by Shweta Amale (shweta.amale@dypic.in)

- Prim's algorithm is used in irrigation channels and placing microwave towers
- It is used in cluster analysis
- Prim's algorithm is used in gaming development and cognitive science
- Path finding algorithms in artificial intelligence and traveling salesman problems make use of prim's algorithm.

VI. Kruskal's Minimal Spanning Tree Algorithm

Kruskal's Algorithm

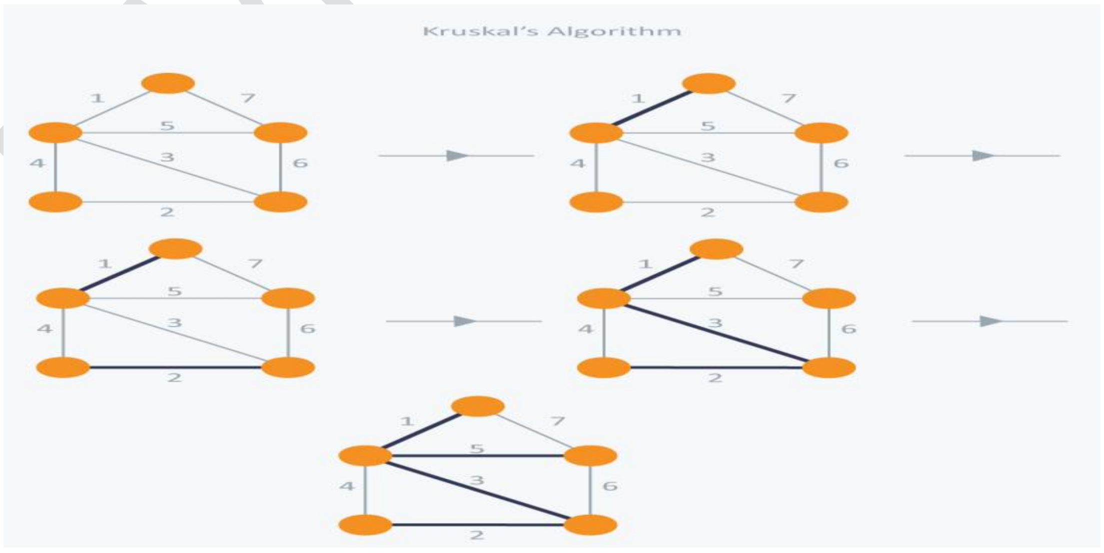
Kruskal's Algorithm builds the spanning tree by adding edges one by one into a growing spanning tree. Kruskal's algorithm follows greedy approach as in iteration it finds edges which has least weight and adds it to the growing spanning tree.

Algorithm Steps:

- Sort the graph edges with respect to their weights.
- Start adding edges to the MST from the edge with the smallest weight until the edge of the largest weight.
- Only add edges which doesn't form a cycle, edges which connect only disconnected components.

So now the question is how to check if 2 vertices are connected or not?
This could be done using DFS which starts from the first vertex, then check if the second vertex is visited or not. But DFS will make time complexity large as it has an order of O(V+E) where V is the number of vertices, E is the number of edges. So the best solution is **"DisjointSets":**
Disjoint sets are sets whose intersection is the empty set so it means that they don't have any element in common.

Consider following example:



This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

In Kruskal’s algorithm, at each iteration we will select the edge with the lowest weight. So, we will start with the lowest weighted edge first i.e., the edges with weight

1. After that we will select the second lowest weighted edge i.e., edge with weight

2. Notice these two edges are totally disjoint. Now, the next edge will be the third lowest weighted edge i.e., edge with weight

3, which connects the two disjoint pieces of the graph. Now, we are not allowed to pick the edge with weight

4, that will create a cycle and we can’t have any cycles. So we will select the fifth lowest weighted edge i.e., edge with weight

5. Now the other two edges will create cycles so we will ignore them. In the end, we end up with a minimum spanning tree with total cost 11 (= 1 + 2 + 3 + 5).

VII. Dijkstra's Minimal Spanning Tree Algorithm

Dijkstra's algorithm has many variants but the most common one is to find the shortest paths from the source vertex to all other vertices in the graph.

Algorithm Steps:

- Set all vertices distances = infinity except for the source vertex, set the source distance = 0.
- Push the source vertex in a min-priority queue in the form (distance, vertex), as the comparison in the min-priority queue will be according to vertices distances.
- Pop the vertex with the minimum distance from the priority queue (at first the popped vertex = source).
- Update the distances of the connected vertices to the popped vertex in case of "current vertex distance + edge weight < next vertex distance", then push the vertex with the new distance to the priority queue.
- If the popped vertex is visited before, just continue without using it.
- Apply the same algorithm again until the priority queue is empty.

Conclusion

As we studied, the minimum spanning tree has its own importance in the real world, it is important to learn the prim’s algorithm which leads us to find the solution too many problems. When it comes to finding the minimum spanning tree for the dense graphs, prim’s algorithm is the first choice.

Staff signature with date

Questions ?

1. What is a Minimum Spanning Tree?
2. What is Prim’s Algorithm?
3. What are the applications for Kruskal’s Algorithm?

MVPS KBTCOE

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Group B
Experiment No: 04

1.1 Aim:
Implement a solution for a Constraint Satisfaction Problem using Branch and Bound and Backtracking for n-queens problem or a graph coloring problem.

1.2 Objectives:
To solve N-queens problem using Constraint Satisfaction Problem

1.3 Software used (if applicable) / Programming Languages Used: 64-bit Windows OS and Linux, C++/, Java, python

1.4. Theory:

What is constraint satisfaction problem?

The idea is that we break the problem up into a set of distinct conditions each of which have to be satisfied for the problem to be solved.

BAKTRACKING

Many problems are difficult to solve algorithmically. Backtracking makes it possible to solve at least some large instances of difficult combinatorial problems. Suppose we have to make a series of decisions among various choices, where

- We don't have enough information to know what to choose
- Each decision leads to a new set of choices.
- Some sequence of choices (more than one choice) may be a solution to your problem.

Applications of Backtracking:

- N Queens Problem
- Sum of subsets problem
- Graph coloring
- Hamiltonian cycles

N-Queens Problem:

A classic combinational problem is to place n queens on a n*n chess board so that no two attack, i.,e no two queens are on the same row, column or diagonal. If we take n=4then the problem is called 4 queens problem. If we take n=8 then the problem is called as 8 queens problem.

Branch and Bound

Branch and Bound is another method to systematically search a solution space. Just like backtracking, we will use bounding functions to avoid generating sub trees that do not contain an answer node.

Branch and Bound differs from backtracking in two important points:

MVPS'S KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

• It has a branching function, which can be a depth first search, breadth first search or based on bounding function.

• It has a bounding function, which goes far beyond the feasibility test as a mean to prune efficiently the search tree.

Branch and Bound refers to all state space search methods in which all children of the E-node are generated before any other live node becomes the E-node.

• Live node is a node that has been generated but whose children have not yet been generated.

• E-node is a live node whose children are currently being explored. In other words, an E-node is a node currently being expanded.

• Dead node is a generated node that is not to be expanded or explored any further. All children of a dead node have already been expanded.

• Branch-an-bound refers to all state space search methods in which all children of an E-node are generated before any other live node can become the E-node.

Graph coloring problem's solution using backtracking algorithm : **Graph coloring**
The **graph coloring problem** is to discover whether the nodes of the graph G can be covered in such a way, that no two adjacent nodes have the same color yet only m colors are used. This graph coloring problem is also known as M-color ability decision problem.
The M – color ability optimization problem deals with the smallest integer m for which the graph G can be colored. The integer is known as a chromatic number of the graph.
Here, it can also be noticed that if d is the degree of the given graph, then it can be colored with d+ 1 color.
A graph is also known to be planar if and only if it can be drawn in a planar in such a way that no two edges cross each other. A special case is the 4 - colors problem for planar graphs. The problem is to color the region in a map in such a way that no two adjacent regions have the same color. Yet only four colors are needed. This is a problem for which graphs are very useful because a map can be easily transformed into a graph. Each region of the map becomes the node, and if two regions are adjacent, they are joined by an edge.
Graph coloring problem can also be solved using a state space tree, whereby applying a backtracking method required results are obtained.
For solving the **graph coloring problem**, we suppose that the graph is represented by its adjacency matrix G[1:n, 1:n], where, G[i, j]= 1 if (i, j) is an edge of G, and G[i, j] = 0 otherwise.

The colors are represented by the integers 1, 2, ..., m and the solutions are given by the n-tuple (x1, x2, x3, ..., xn), where x1 is the color of node i.

Conclusion

Here we have implemented a solution for a Constraint Satisfaction Problem using Branch and Bound and Backtracking for n-queens problem or a graph coloring problem.

Staff signature with date

Question?

1. Explain 8 Queens Problem using Branch and Bound
2. Explain Graph coloring problem's solution using backtracking algorithm

MVPS KBTCOE

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Group B
Experiment No: 05

1.1 Aim:
Develop an elementary chatbot for any suitable customer interaction application.

1.2 Objectives:
To develop an elementary chatbot for any suitable customer interaction application

1.3 Software used (if applicable) / Programming Languages Used: 64-bit Windows OS and Linux, C++/, Java, python

1.4. Theory:

What is a chatbot?
A chatbot is a computer program designed to have a conversation with human beings over the internet. It's also known as conversational agents, which communicate and collaborate with human users, through text messaging, in order to accomplish a specific task. Basically, there are two types of chatbots. The one that uses **Artificial Intelligence** and another one is based on multiple choice scripts.
Both types of chatbots aim to create a more personalized content experience for the users, whether that's while watching a video, reading articles or buying new shoes.
These Chatbots hold the promise of being the next generation of technology that people use to interact online with business enterprises. These Chatbots offer a lot of advantages, one of which is that, because Chatbots communicate using a natural language, users don't need to learn yet another new website interface, to get comfortable with the unavoidable quirks.
Chatbots are capable to interpret human speech, and decide which information is being sought. Artificial intelligence is getting smarter each day, and brands that are integrating Chatbots with the artificial intelligence, can deliver one-to-one individualized experiences to consumers.

Why chatbot?
Chatbots can be useful in many aspects of the customer experience, including providing customer service, presenting product recommendations and engaging customers through targeted marketing campaigns. If a customer has an issue with a

MVPS KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

product, she can connect with a chatbot to explain the situation and the chatbot can input that information to provide a recommendation of how to fix the product. On the recommendation side, chatbots can be used to share popular products with customers that they might find useful and can act as a sort of personal shopper or concierge service to find the perfect gift, meal or night out for a customer with just a few basic questions. Brands are also using chatbots to connect their customers with thought leaders and add personality to their products. In all cases, brands seem to be having great success and experiencing increased engagement and revenue.

Chatbots are easy to use and many customers prefer them over calling a representative on the phone because it tends to be faster and less invasive. They can also save money for companies and are easy to set up.

Chatbots are relatively new and most companies haven't implemented them yet, it's only natural that users are interested in them. Hence, people want to discover what chatbots can and cannot do.

The number of businesses using chatbots has grown exponentially. Chatbots have increased from 30,000 in 2016 to over 100,000 today. Every major company has announced their own chatbot and 60% of the youth population uses them daily.

These statistics prove that chatbots are the new-gen tech. No more waiting for the right time to incorporate them into your business. The time is now. By the year 2020, nearly 80% of businesses will have their own chatbot.

Billions of people are already using chatbots, so it's time your business did too.

Benefits of chatbot?

Chatbots are being made to ease the pain that the industries are facing today. The purpose of chat bots is to support and scale business teams in their relations with customers.

Chatbots may sound like a futuristic notion, but according to Global Web Index statistics, it is said that 75% of internet users are adopting one or more messenger platforms. Although research shows us that each user makes use of an average of 24 apps a month, wherein 80% of the time would be in just 5 apps. This means you can hardly shoot ahead with an app, but you still have high chances to integrate your chatbot with one of these platforms.

Benefits that chatbots provide:

1. Available 24*7:

I'm sure most of you have experienced listening to the boring music playing while you're kept on hold by a customer care agent. On an average people spend 7 minutes

MVPS KBTCOE

Downloaded by Shweta Amale (shweta.amale@dypic.in)

until they are assigned to an agent. Gone are the days of waiting for the next available operative. Bots are replacing live chat and other forms of contact such as emails and phone calls.
Since chat bots are basically virtual robots they never get tired and continue to obey your command. They will continue to operate every day throughout the year without requiring to take a break. This improves your customer satisfaction and helps you rank highly in your sector.

2. Handling Customers:

We humans are restricted to the number of things we can do at the same time. A study suggests that humans can only concentrate on 3–4 things at the same time. If it goes beyond that you are bound to meet errors.
Chatbots on the other hand can simultaneously have conversations with thousands of people. No matter what time of the day it is or how many people are contacting you, every single one of them will be answered instantly. Companies like Taco Bell and Domino's are already using chatbots to arrange delivery of parcels.

3. Helps you Save Money:

If you are a business owner you are bound have a lot of employees who need to be paid for the work they do. And these expenses just keep adding up as business grows. Chatbots are a one time investment which helps businesses reduce down on staff required.
You could integrate a customer support chatbot in your business to cater to simple queries of customers and pass on only the complex queries to customer support agents.

4. Provides 100% satisfaction to customers:

Humans react to others based on their mood and emotions. If a agent is having a good attitude or is in good mood he will most probably talk to customers in a good way. In contrary to this the customer will not be satisfied.
Whereas chatbots are bound by some rules and obey them as long as they're programmed to. They always treat a customer in the most polite and perfect way no matter how rough the person is. Also, in the travel and hospitality industry where travelers do not speak the same language, a bot can be trained to communicate in the language of the traveler.

5. Automation of repetitive work:

MVPS KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Lets be honest, no one likes doing the same work again and again over brief period of time. In the case of humans, such tasks are prone to errors. Chatbots now help automate tasks which are to be done frequently and at the right time.
Also, now there are numerous slack bots which automate repetitive tasks. This helps people save time and increase productivity. For example, there are new items bought from your eCommerce site or there is a bug reported then it sends a short summary to a slack channel.

6. Personal Assistant:

People could use Bots as a fashion advisor for clothing recommendations, or ask trading tips from a finance bot, suggest places to visit from a travel bot and so forth. This would help the users get a more personal touch from the chatbot. Also, the chatbot will remember all your choices and provide you with relevant choices the next time you visit it.

How chatbot can drive revenue for you?

Below we have compiled reasons why chatbots are important for your business and how can they help in increasing revenues:
a. Higher user customer engagement

Most businesses these days have a web presence. But with being on the internet, boundaries of day and night, availability and unavailability have changed, so have user expectations. This is probably the biggest reason to use them. Bots give the user an interactive experience. It makes customers feel they are working with someone to help resolve their issue. If done right, bots can help customers find what they are looking for and make them more likely to return.

Customer Engagement

 Clearance Sale : Notify users about on-going clearance sale of products relevant to the users at their nearest outlets.

 Product Finder : Enable consultative selling without the need of a call center

 It offer Notification : Notify users about offers, product launches on products/ services they've shown interest in, and products that's back in stock

b. **Mobile-ready and immediate availability**

Along with a web presence, it has also become increasingly important for brands to have a mobile presence - mobile apps, mobile-optimized websites. Considering how chat has been around on the mobile for ages, most chatbot

MVP'S KBTCOE

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Implementations don't need you to work on tweaking their UI, they are ready to implement and so available to your customers immediately
You might argue that you have an app for that. Having an app for your brand is great, but having users discover that app, download it and use it to stay engaged is not an easy deal. Instead, implementing a chatbot - which works on the mobile browser or a messaging-app which the user regularly uses - makes it all the more reason for a customer to be engaged with the brand
c. It can drive sales

Chatbots can be intelligent. Depending on a user's preferences or purchases, it can send products to customers which are more likely to convert into sales. Or it can send coupons to users for in-store purchases/discounts. Bots can also be used to link the user to your mCommerce site/app so they can buy the product directly from the convenience of their phones
Sell Intelligently
 Product Recommendations: Push proactive recommendations to users based on their preferences and search and order history.
 Enable order booking over chat.

d. **Minimal cost - Maximum return**

The best part about bots is they are cheap. Chatbot provide the necessary infrastructure and APIs for creating these bots. They require minimal maintenance and since it is automated, there is no labor-intensive work that goes in there.
e. Customer Service

 Track Order: Keep users up to date with order status. Schedule or reschedule delivery to a provided address or request to pick it up at any other Best Buy outlet.

 Stock outs: Notify users when desired product is available and place order over a chat.

 Returns and Replacements: No waiting time to reach customer care. Customers can instantly place request to replace or return an order.

 Seek Reviews: Reach out to users to seek reviews on the products recently bought

Gift Recommendations

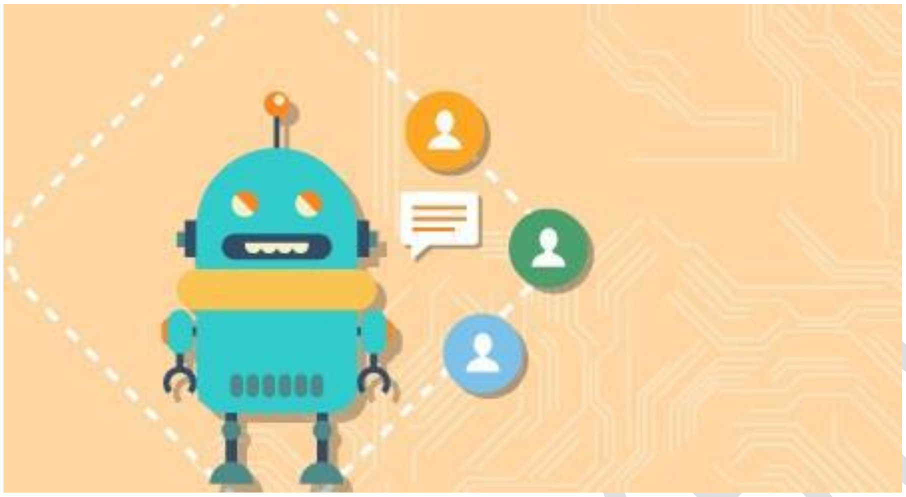
MVPS's KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

 Recommend relevant gifting options to users, accessing calendar events and understanding the likes and style of beneficiary.
 Opportunity to upsell gift cards for the users for every occasion.

Application across Industries



According to a new survey, 80% of businesses want to integrate chatbots in their business model by 2020. So which industries can reap the greatest benefits by implementing consumer-facing chatbots? According to a chatbot, these major areas of direct-to-consumer engagement are prime:

Chatbots in Restaurant and Retail Industries

Famous restaurant chains like Burger King and Taco bell have introduced their Chatbots to stand out of competitors of the Industry as well as treat their customers quickly. Customers of these restaurants are greeted by the resident Chatbots, and are offered the menu options- like a counter order, the Buyer chooses their pickup location, pays, and gets told when they can head over to grab their food. Chatbots also works to accept table reservations, take special requests and go take the extra step to make the evening special for your guests.

Chatbots are not only good for the restaurant staff in reducing work and pain but can provide a better user experience for the customers.

Chatbots in Hospitality and Travel

For hoteliers, automation has been held up as a solution for all difficulties related to productivity issues, labour costs, a way to ensure consistently, streamlined production processes across the system. Accurate and immediate delivery of information to customers is a major factor in running a successful online Business, especially in the price sensitive and competitive Travel and Hospitality industry. Chatbots particularly have gotten a lot of attention from the hospitality industry in recent months.

MVPS KBTCOE

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Chatbots can help hotels in a number of areas, including time management, guest services and cost reduction. They can assist guests with elementary questions and requests. Thus, freeing up hotel staff to devote more of their time and attention to time-sensitive, critical, and complicated tasks. They are often more cost effective and faster than their human counterparts. They can be programmed to speak to guests in different languages, making it easier for the guests to speak in their local language to communicate.

Chatbots in Health Industry

Chatbots are a much better fit for patient engagement than Standalone apps. Through these Health-Bots, users can ask health related questions and receive immediate responses. These responses are either original or based on responses to similar questions in the database. The impersonal nature of a bot could act as a benefit in certain situations, where an actual Doctor is not needed.

Chatbots ease the access to healthcare and industry has favourable chances to serve their customers with personalised health tips. It can be a good example of the success of Chatbots and Service Industry combo.

Chatbots in E-Commerce

Mobile messengers- connected with Chatbots and the E-commerce business can open a new channel for selling the products online. E-commerce Shopping destination "Spring" was the early adopter. E-commerce future is where brands have their own Chatbots which can interact with their customers through their apps.

Chatbots in Fashion Industry

Chatbots, AI and Machine Learning pave a new domain of possibilities in the Fashion industry, from Data Analytics to Personal Chatbot Stylists. Fashion is such an industry where luxury goods can only be bought in a few physical boutiques and one to one customer service is essential. The Internet changed this dramatically, by giving the customers a seamless but a very impersonal experience of shopping. This particular problem can be solved by Chatbots. Customers can be treated personally with bots, which can exchange messages, give required suggestions and information. Famous fashion brands like Burberry, Tommy Hilfiger have recently launched Chatbots for the London and New York Fashion Week respectively. Sephora a famous cosmetics brand and H&M– a fashion clothing brand have also launched their Chatbots.

MVPS KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Chatbots in Finance

Chatbots have already stepped in Finance Industry. Chatbots can be programmed to assists the customers as Financial Advisor, Expense Saving Bot, Banking Bots, Tax bots, etc. Banks and Fintech have ample opportunities in developing bots for reducing their costs as well as human errors. Chatbots can work for customer’s convenience, managing multiple accounts, directly checking their bank balance and expenses on particular things.

Chatbots in Fitness Industry

Chat based health and fitness companies using Chatbot, to help their customers get personalised health and fitness tips. Tech based fitness companies can have a huge opportunity by developing their own Chatbots offering huge customer base with personalised services. Chatbots and Service Industry together have a wide range of opportunities and small to big all size of companies using chatbots to reduce their work and help their customers better.

Chatbots in Media

Big publisher or small agency, our suite of tools can help your audience chatbot experience rich and frictionless. Famous News and Media companies like The Wall Street Journal, CNN, Fox news, etc have launched their bots to help you receive the latest news on the go. **Chatbot in Celebrity:**

With a chatbot you can now have one-on-one conversation with millions of fans.

Chatbot in Marketing

SMS Marketing

 Why promote just a coupon code that the customer does not know how to use?

 Improve conversions from existing SMS campaigns

 Talk to your customers when they want to using “Talk to an Agent” feature.

MVPS'S KBTCOE

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Email Marketing
 so your eMail has made a solid elevator pitch about your product.
 As a next step, is making customers fill an online form the most exciting way to engage with your customers?
 It's time to rethink the landing page.
 Instantly engage in a conversation with your customers.
 Address their concerns and queries

Social Media Triage
 How effectively are you addressing the negative sentiment around your brand on social media?
 Addressing queries instantly and effectively can convert even an angry customer into a loyal fan.
 Leverage a chatbot as your first response strategy and comfort that customer

Process
Stage #1: Chatty Bot welcomes you Teach your assistant to introduce itself in the console.
Stage #2: Print your name Introduce yourself to the bot.
Stage #3: Guess the age Use your knowledge of strings and numbers to make the assistant guess your age.
Stage #4: Learning numbers Your assistant is old enough to learn how to count. And you are experienced enough to apply a for loop at this stage!
Stage #5: Multiple Choice At this point, the assistant will be able to check your knowledge and ask multiple-choice questions. Add some functions to your code and make the stage even better.

How To Run The Project?
To run this project, you must have installed Python on your PC. After downloading the project, follow the steps below:
Step1: Extract/Unzip the file
Step2: Go inside the project folder, open cmd then type bot.py and enter to start the system.
OR
Step2: Simply, double-click the bot.py file and you are ready to go.

Conclusion

Hence we have developed an elementary chatbot for customer interaction application

Staff signature with date

Questions

1. What is Chatbot? Why chatbot is required?
2. Explain Benefits of Chatbot?
3. Explain applications of chatbot

MVPS'S KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Group C
Experiment No: 06

1.1 Aim:

Implement any one of the following Expert System

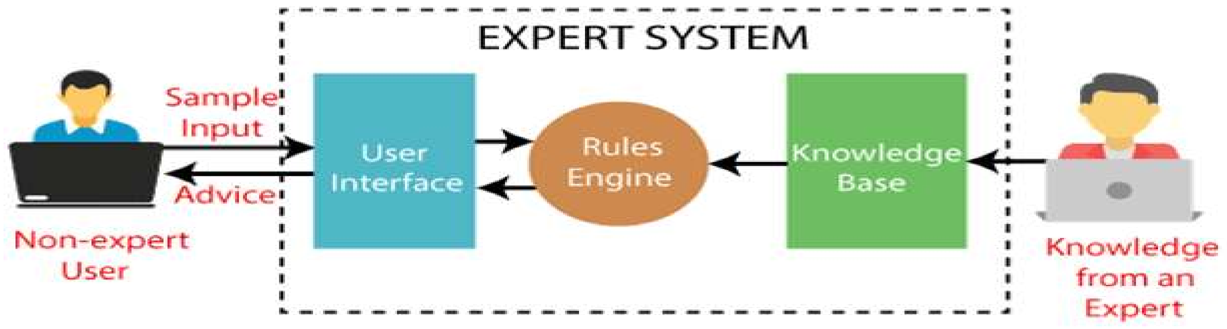
1. Information management
2. Hospitals and medical facilities
3. Help desks management
4. Employee performance evaluation
5. Stock market trading
6. Airline scheduling and cargo schedules

1.2 Objectives:
To develop an elementary chatbot for any suitable customer interaction application

1.3 Software used (if applicable) / Programming Languages Used: 64-bit Windows OS and Linux, C++/, Java, python

1.4. Theory:

What is an Expert System?
An expert system is a computer program that is designed to solve complex problems and to provide decision-making ability like a human expert. It performs this by extracting knowledge from its knowledge base using the reasoning and inference rules according to the user queries.
The expert system is a part of AI, and the first ES was developed in the year 1970, which was the first successful approach of artificial intelligence. It solves the most complex issue as an expert by extracting the knowledge stored in its knowledge base. The system helps in decision making for compsex problems using **both facts and heuristics like a human expert**. It is called so because it contains the expert knowledge of a specific domain and can solve any complex problem of that particular domain. These systems are designed for a specific domain, such as **medicine, science,** etc.
The performance of an expert system is based on the expert's knowledge stored in its knowledge base. The more knowledge stored in the KB, the more that system improves its performance. One of the common examples of an ES is a suggestion of spelling errors while typing in the Google search box.
Below is the block diagram that represents the working of an expert system:



Downloaded by Shweta Amale (shweta.amale@dypic.in)

ANDPS KBTCOE

Below are some popular examples of the Expert System:

o **DENDRAL:** It was an artificial intelligence project that was made as a chemical analysis expert system. It was used in organic chemistry to detect unknown organic molecules with the help of their mass spectra and knowledge base of chemistry.

o **MYCIN:** It was one of the earliest backward chaining expert systems that was designed to find the bacteria causing infections like bacteraemia and meningitis. It was also used for the recommendation of antibiotics and the diagnosis of blood clotting diseases.

o **PXDES:** It is an expert system that is used to determine the type and level of lung cancer. To determine the disease, it takes a picture from the upper body, which looks like the shadow. This shadow identifies the type and degree of harm.

o **CaDeT:** The CaDet expert system is a diagnostic support system that can detect cancer at early stages.

Characteristics of Expert System

o **High Performance:** The expert system provides high performance for solving any type of complex problem of a specific domain with high efficiency and accuracy.

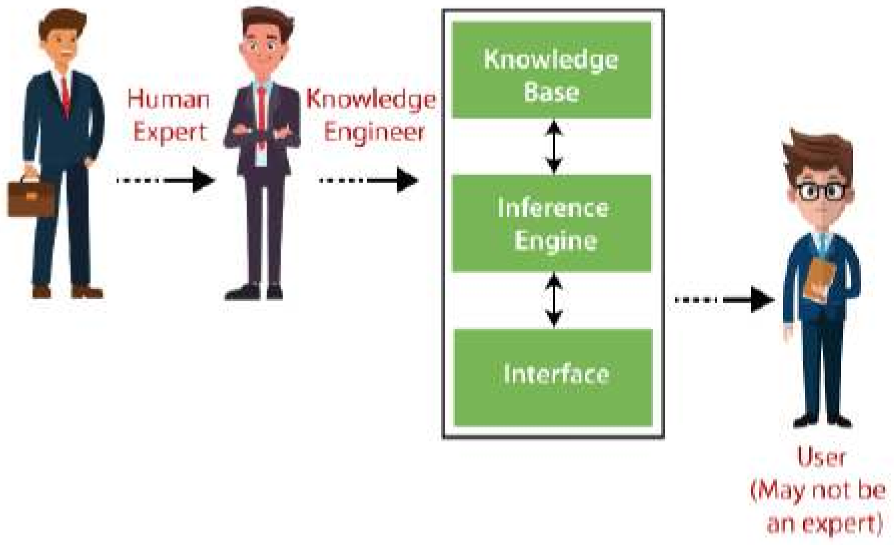
o **Understandable:** It responds in a way that can be easily understandable by the user. It can take input in human language and provides the output in the same way.

o **Reliable:** It is much reliable for generating an efficient and accurate output.

o **Highly responsive:** ES provides the result for any complex query within a very short period of time.

Components of Expert System

An expert system mainly consists of three components:



This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)

- **User Interface**
- **Inference Engine**
- **Knowledge Base**

Participants in the development of Expert System
There are three primary participants in the building of Expert System:
1. **Expert:** The success of an ES much depends on the knowledge provided by human experts. These experts are those persons who are specialized in that specific domain.

2. **Knowledge Engineer:** Knowledge engineer is the person who gathers the knowledge from the domain experts and then codifies that knowledge to the system according to the formalism.

3. **End-User:** This is a particular person or a group of people who may not be experts, and working on the expert system needs the solution or advice for his queries, which are complex.

Advantages of Expert System
- These systems are highly reproducible.
- They can be used for risky places where the human presence is not safe.
- Error possibilities are less if the KB contains correct knowledge.
- The performance of these systems remains steady as it is not affected by emotions, tension, or fatigue.
- They provide a very high speed to respond to a particular query.

Limitations of Expert System
- The response of the expert system may get wrong if the knowledge base contains the wrong information.
- Like a human being, it cannot produce a creative output for different scenarios.
- Its maintenance and development costs are very high.
- Knowledge acquisition for designing is much difficult.
- For each domain, we require a specific ES, which is one of the big limitations.
- It cannot learn from itself and hence requires manual updates.

Applications of Expert System
In designing and manufacturing domain It can be broadly used for designing and manufacturing physical devices such as camera lenses and automobiles.

In the knowledge domain these systems are primarily used for publishing the relevant knowledge to the users. The two popular ES used for this domain is an advisor and a tax advisor.

In the finance domain in the finance industries, it is used to detect any type of possible fraud, suspicious activity, and advise bankers that if they should provide loans for business or not.

In the diagnosis and troubleshooting of devices in medical diagnosis, the ES system is used, and it was the first area where these systems were used.

MVPS's KBTCOE

Downloaded by Shweta Amale (shweta.amale@dypic.in)

Planning and scheduling The expert systems can also be used for planning and scheduling some particular tasks for achieving the goal of that task.

Staff signature with date

Conclusion

Hence we have implemented the expert system

Question

1. What is expert system. Describe it in detail?
2. Explain Advantages of Expert System with their limitations?
3. Enlist application of expert system?

MVPS'S KBTCOE

This document is available free of charge on studocu

Downloaded by Shweta Amale (shweta.amale@dypic.in)