IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 10, October 2021

DOI: 10.17148/IJARCCE.2021.101020

Evaluation of Standard Greedy Approach for Networking Algorithms

Barkha Gupta

Programme Assistant, Department of Computer, Agriculture University, Jodhpur, Rajasthan, India.

Abstract: Algorithm in general terms is set of rules to be followed for doing a calculation. In computer science, an algorithm is a finite set of steps of well-defined instructions to perform a particular task or to solve a well-defined problem. Various type of algorithm is developed and used in the field of networking. Some may be divide and conquer algorithm other may follows the standards of dynamic programming and some may follow the greedy approach.
Greedy is an algorithmic paradigm that builds up a solution piece by piece. Greedy algorithm always chooses the piece that offers the most obvious and immediate and nearest benefit. So, this problem was choosing locally optimal also leads to globally solution that is best fit for Greedy.

Keywords: Knapsack problem, Standard greedy algorithms, Graph map colouring, Graph vertex cover, Kruskal algorithm, Prim's Algorithm, Job scheduling.

I. INTRODUCTION

Greedy Algorithm is an algorithm that solves the problem by locally choosing the optimal choice at each stage with a goal to achieve global optimal solution. This algorithm follows the heuristic solution at each stage and might fails sometimes to provide globally optimum solution as choosing optimum at each stage is not definitely correct every time. The very basic problem is counting coins while using minimum number of coins used. Let say we have multiple coins of 1, 7, 10 rupees. Now we have purchased a book of value 15. So, for giving the value 15 we have various options.
Normal option of making value 15 is

Coin of 10	Coin of 1	Coin of 1	Coin of 1	Coin of 1	Coin of 1	Total coins used = 6

Greedy option of using minimum coins is

Coin of 7	Coin of 7	Coin of 1	Total coins used = 3

II. TYPES OF PROBLEMS

(i) Minimization Problem - A problem that demands a minimum result is known as the minimization problem. Minimization problem cannot be solved by the corner point method.

(ii) Maximization Problem - A problem that demands a maximum result is known as the Maximization problem. Maximization problem can be solved by the corner point method.

(iii) Optimization Problem – A problem which is in search of finding the best solution from all the feasible solution is known as optimization problem.

III. TYPES OF SOLUTIONS

(i) Feasible Solutions - Whenever a problem arises, one has many possible solutions to that problem. Yet, taking into consideration the condition set on that problem, we choose solutions that satisfy the given condition. Such solutions that help us to get the result which meets the given condition is known as Feasible Solution**.**

(ii) Optimal Solutions- A solution is called optimal when it is already feasible and satisfy all the constraints given. It achieves the objective of the problem and which is the best result. This objective could either be the minimum or maximum result. The point to be considered here is that any problem will only have one optimal solution.

IV. COMPONENTS OF GREEDY ALGORITHM

Few components of greedy algorithms are: -

(i) Candidate set – A set from which output/solution is created

(ii) Selection function – It is used to select the best candidate that is to be added to the solution

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 10, October 2021

DOI: 10.17148/IJARCCE.2021.101020

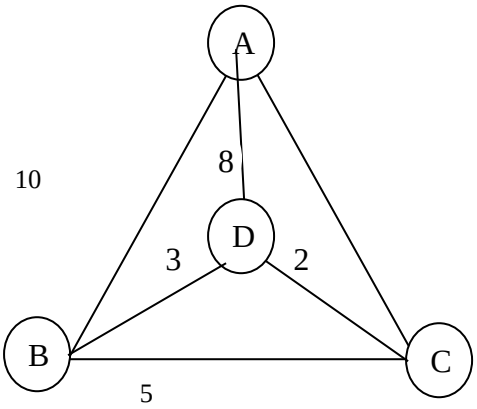
(iii) Feasibility function – It is used to find out whether a candidate can be used to contribute to a solution

(iv) Objective function – It is used to assigns a value to a solution, or a partial solution or an intermediate solution.

(v) Solution function – It is used to find out that whether we reached to a complete solution or not.

V. TRAVELLING SALESMAN PROBLEM

Travelling Salesman Problem is problem where a person has to start from one city and visit all the other cities exactly once. Distance between the cities is already known in prior and visitor has to return back to its origin city from where he has started by visiting all cities exactly once with a condition to find out the shortest route between the cities.



Possible solution is as follows: -
A, B, C, D are known as nodes and here it implies cities.
10, 8, 6, 3, 2, 5 are the weightage of the edges.

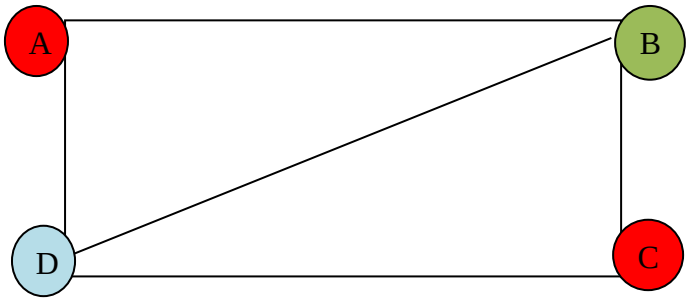
Route	Origin City	Hop 1	Hop 2	Hop 3	Hop 4/Back to origin	Calculating
Route 1	B	C	D	A	B	5+2+8+10=25
Route 2	B	C	A	D	B	5+6+8+3=22
Route 3	B	D	C	A	B	3+2+6+10=21

So, as per Travelling Salesman Problem Route 3 is the optimum solution. Hence, this route will be selected.

VI. GRAPH MAP COLORING

Graph map colouring is a type of greedy algorithm. In which each vertex of the graph is been coloured with any chosen colour. The primary condition is no adjacent edges have same colour of their respective vertices. The objective is to minimize the number of colours while colouring a graph. The smallest number of colours required to colour a graph G is called its chromatic number of that graph.

Example 1:



In above example 1, 3 colours are required to colour a graph. **Hence, its chromatic number is = 3.**

IJARCCE

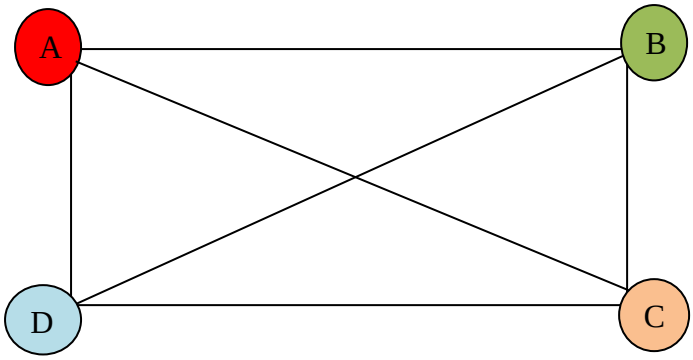
ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 10, October 2021

DOI: 10.17148/IJARCCE.2021.101020

Example 2:



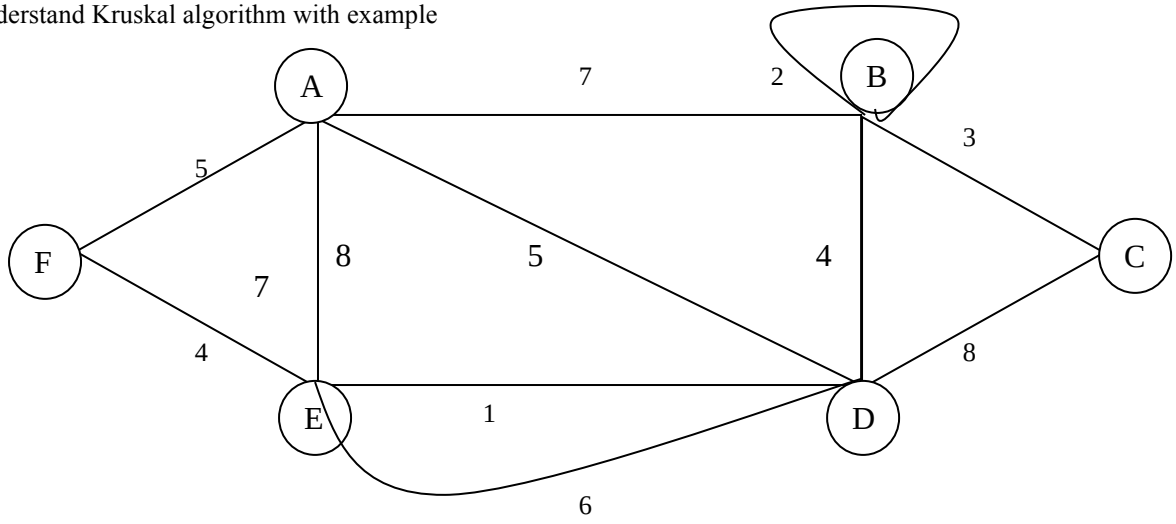
In above example 2, 4 colours are required to colour a graph because all the edges are adjacent to each other.
Hence, its chromatic number is = 4.

VII. KRUSKAL ALGORITHM

Kruskal Algorithm is an algorithm used to find out minimum cost spanning tree. It uses the greedy approach. It also follows the properties of spanning tree. Few of the properties of spanning tree are mentioned below: -

(i) Spanning tree is a graph which is connected.
(ii) All the vertices should be covered.
(iii) Spanning tree does not contain any cycle
(iv) A connected graph can have more than one spanning trees.
(v) Spanning Tree has (n-1) edges where n is the number of vertices.
(vi) If all the edge weights of a given graph are the same, then every spanning tree of that graph is minimum.
(vii) If each edge has a distinct weight, then there will be only one, unique minimum spanning tree.

Let's understand Kruskal algorithm with example



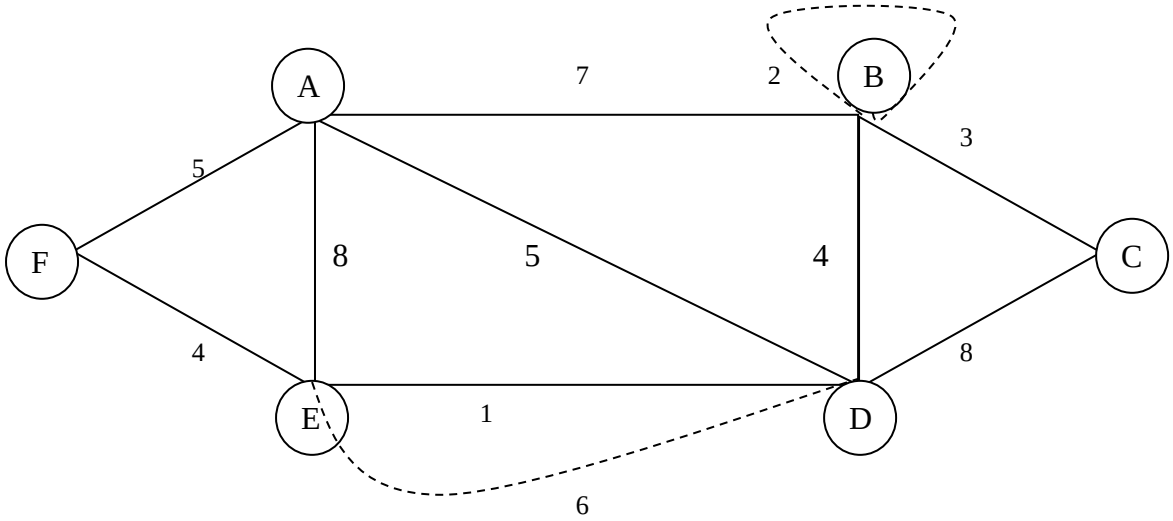
IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 10, October 2021

DOI: 10.17148/IJARCCE.2021.101020

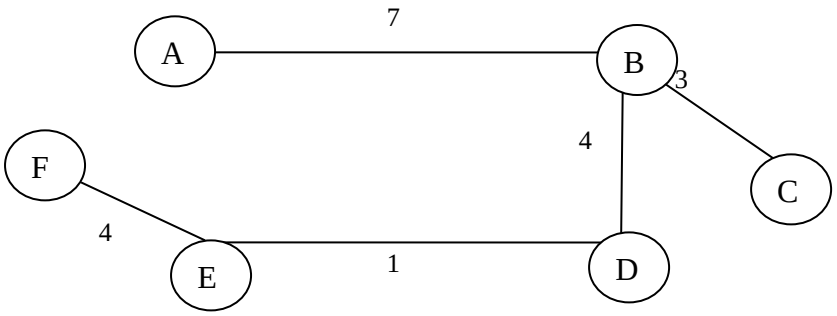
Step 1: Remove all the parallel edges (keep the edge with minimum weight associated with the edges) and loops.



Step 2: Arrange all the edges in the ascending order in terms of their weight associated with their edges

S. No.	Edges pair	Weight Associated
1	D, E	1
2	B, C	3
3	E, F	4
4	B, D	4
5	A, D	5
6	A, F	5
7	A, B	7
8	A, E	8
9	C, D	8

Step 3: Now create a graph by adding the edges with minimum weightage first.



We cannot add edges (A, E) and (A, D) because they will make a loop inside the graph which is not a property of spanning tree.

VIII. PRIMS ALGORITHM

Prim's Algorithm is an algorithm used to find out minimum cost spanning tree. It also uses the greedy approach. It follows all the properties of spanning tree (mentioned in Kruskal algorithm). Difference in Kruskal and prim's algorithm is that Kruskal goes for adding the minimum edges in the graph and prim's starts from an origin node and choose the next node with the minimum cost.

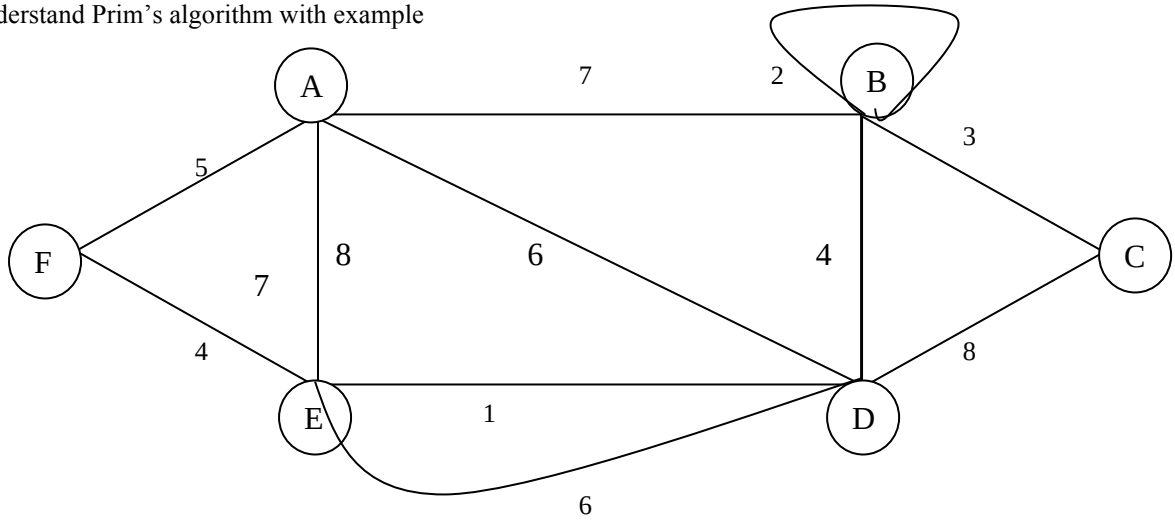
IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

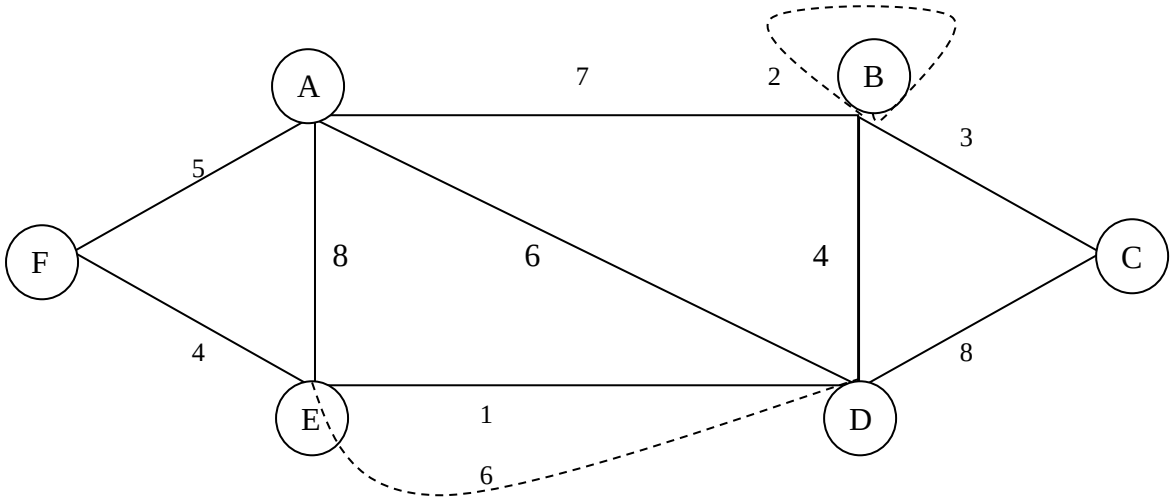
Vol. 10, Issue 10, October 2021

DOI: 10.17148/IJARCCE.2021.101020

Let's understand Prim's algorithm with example



Step 1: Remove all the parallel edges (keep the edge with minimum weight associated with the edges) and loops.

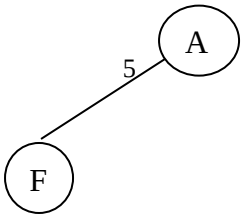


Step 2: choose any arbitrary node as a starting/origin node. Here let assume A as the stating node.

Step 3: Now start from A node and choose its neighbour node with the less cost edge. So, here from A node following edges are available

S. No.	Edges pair	Weight Associated
1	A, E	8
2	A, B	7
3	A, D	6
4	A, F	5

So, minimum cost edge from A node is A, F



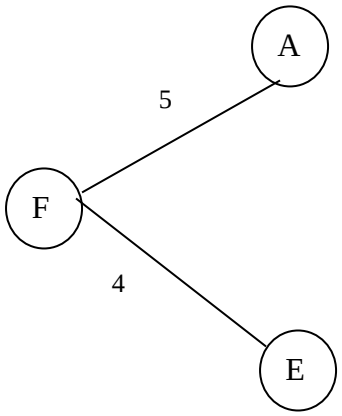
IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 10, October 2021

DOI: 10.17148/IJARCCE.2021.101020

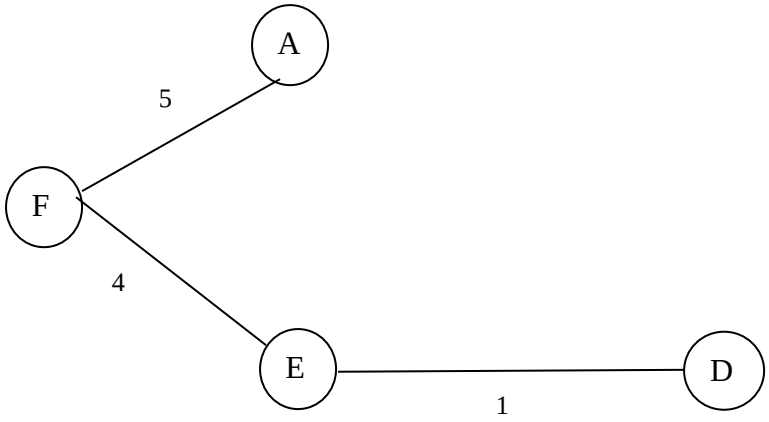
Step 4: Now from F node and choose its neighbour node with the less cost edge. So, here from F node one neighbour node exist that is (F, E)



Step 5: Now from E node and choose its neighbour node with the less cost edge. So, here from E node following edges are available

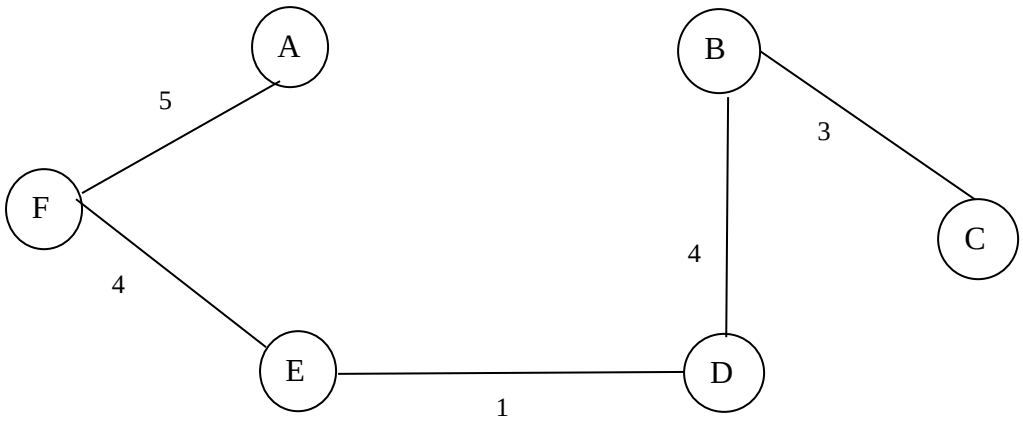
S. No.	Edges pair	Weight Associated
1	A, E	8
2	E, D	1

So, minimum cost edge from A node is E, D



And so, on the final output will be

Step 6: And so, on the final output will be



IJARCCE

ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

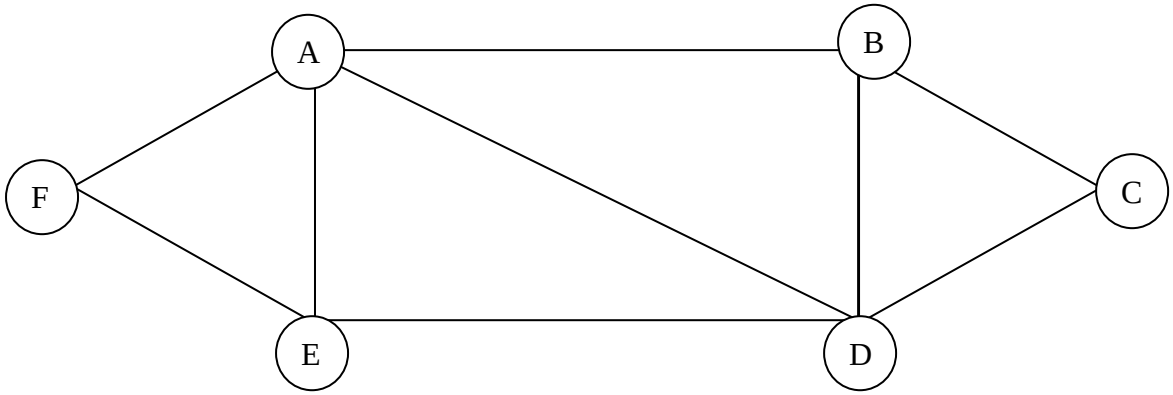
Vol. 10, Issue 10, October 2021

DOI: 10.17148/IJARCCE.2021.101020

IX. GRAPH VERTEX COVER

A vertex cover is a graph which cover all the edges of a given graph. It is an undirected graph G such that for every edge (u, v) of a graph either 'u' or 'v' is in the vertex cover. In an undirected graph, the vertex cover problem is used to find out the minimum size vertex cover. It uses the greedy approach.
Vertex Cover Problem is also known as NP Complete problem, which means that there is no polynomial-time solution for this unless P = NP. There are approximate polynomial-time algorithms to solve this problem.

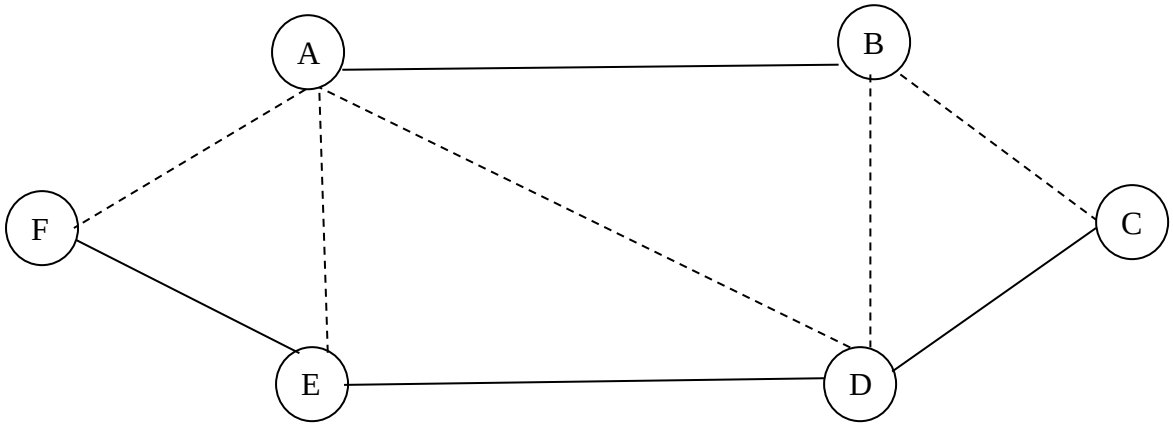
Let's understand vertex cover problem with an example



Step 1: Set of edges in the above graph are

Set	Edges
Set 1	A, B
Set 2	A, D
Set 3	A, E
Set 4	A, F
Set 5	D, E
Set 6	B, D
Set 7	B, C
Set 8	C, D
Set 9	E, F

Step 2: choose an arbitrary edge to start with. Let we start with edge (A, B) and now remove all the edges which is connected to A or B.



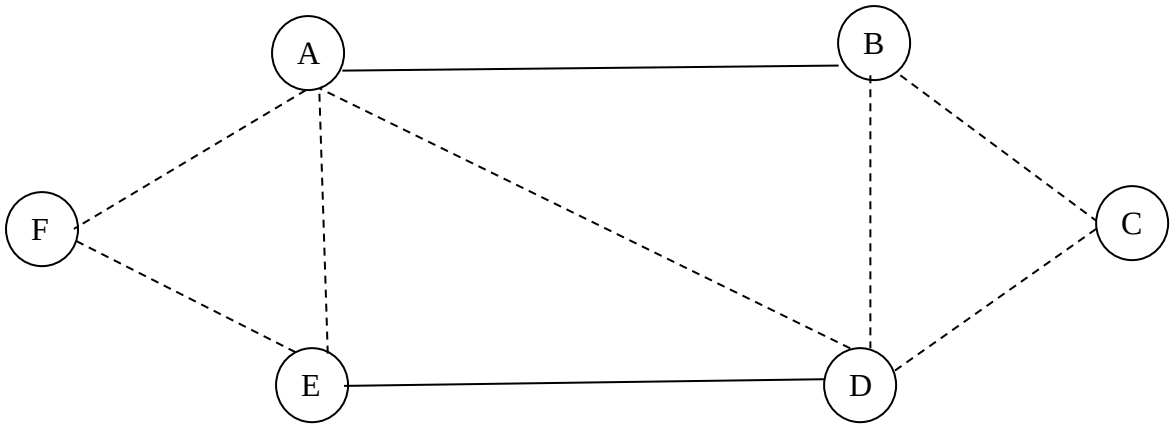
IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 10, October 2021

DOI: 10.17148/IJARCCE.2021.101020

Step 3: choose any other edge from remaining on random basis. Let we choose D, E and now remove all the edges which is connected to D or E.



Step 4: Hence the vertex cover of this graph is {A, B, D, E}

X. JOB SCHEDULING ALGORITHM

Job scheduling algorithm is containing an array of the job that is to be completed with the deadline which is already given with an objective to generate maximum profit. This algorithm is also known as job sequencing algorithm. Thus, in job scheduling algorithm objective is to find out sequence of the job which is completed within their deadline given and generated maximum profit. This is a standard greedy algorithm. The time complexity of this solution is exponential. In this algorithm we will arrange the job in the descending order as per their profit given.

Let's understand job scheduling problem with an example

Job	**J_1**	**J_2**	**J_3**	**J_4**	**J_5**	**J_6**	**J_7**	**J_8**
Deadline	2	1	3	2	1	4	1	3
Profit	60	100	20	40	20	80	70	10

Step1: Arrange the job given in the descending order of their profit given

Job	**J_2**	**J_6**	**J_7**	**J_1**	**J_4**	**J_3**	**J_5**	**J_8**
Deadline	1	4	1	2	2	3	1	3
Profit	100	80	70	60	40	20	20	10

Step 2: Now select the job with maximum profit and that can be completed within the deadline. So, here J_2 job is selected with maximum profit of 100 within deadline of 1 and now no job with same deadline will be selected. So, as per this J_7 and J_5 cannot be completed within their deadlines.

Step 3: Now select the next job with remaining maximum profit and that can be completed within the deadline. So, here J_6 job is selected with second maximum profit of 80 within deadline of 4 and now no job with same deadline will be selected.

Step 4: Now select the next job with remaining maximum profit and that can be completed within the deadline. So, here J_1 job is selected with maximum profit of 60 within deadline of 2 and now no job with same deadline will be selected. So, as per this J_4 cannot be completed within their deadline.

Step 5: Now select the next job with remaining maximum profit and that can be completed within the deadline. So, here J_3 job is selected with maximum profit of 20 within deadline of 3 and now no job with same deadline will be selected. So, as per this J_8 cannot be completed within their deadline.

Step 6: So, the sequence of the job will be (J_2, J_6, J_1, J_3) with the maximum profit of (100+80+60+20) = 260 within their deadlines.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 10, October 2021

DOI: 10.17148/IJARCCE.2021.101020

XI. KNAPSACK PROBLEM

Knapsack basically means a bag or understand a luggage bag. Knapsack problem is given with two array value [1…100] and weight [1…100]. Value implies value of the item and weight is associated with its value. Maximum weightage of the bag (W) is also given. This is a standard greedy algorithm. The objective of knapsack algorithm is to find out the maximum value subset of value [ ] such that sum of the weights of this subset is smaller than or equal to Weightage of bag (W).

Let's understand knapsack problem with an example
Maximum capacity of the sack is 8 and find out the maximum value within the given weight. Value and its associated weight are given below.

Value (V)	10	18	5	25	15
Weight (W)	5	2	6	4	1

Arrange them in the ascending order of their weight

Value (V)	15	18	25	10	5
Weight (W)	1	2	4	5	6

Weight Limit	0	1	2	3	4	5	6	7	8
V = 15, W=1	0	15	15	15	15	15	15	15	15
V=18, W=2	0	15	18	33	33	33	33	33	33
V=25, W=4	0	15	18	33	25	40	43	58	58
V= 10, W=5	0	15	18	33	25	10	25	28	43
V=5, W= 6	0	15	18	33	25	10	5	20	23

So, the maximum weight of the sack is 58

XII. CONCLUSION

Greedy algorithm is the collection of algorithms that have one common characteristics in it. It basically chooses the best choice at each step without considering the future aspects. Thus, the essence of greedy algorithm is choice function. It totally depends on the choice made at each step which depends on the current situation and the current option given.
Greedy algorithm is best applicable when one needs a solution in real-time and approximate answers are "good enough". Clearly, it minimizes the time while making sure that an optimal solution is produced; hence it is more applicable to use in a situation where less time is required. Now, everyone has an idea when to apply this algorithm as it is the best framework that answers nearly all programming challenges along with helping you to decide the most optimal solution at a given point of time.

XIII. ACKNOWLEDGEMENTS

Author is grateful to Agriculture University, Jodhpur for encouraging writing in my field.

XIV. REFERENCES

[1] Narasimha Karumanchi. Algorithm Design Techniques: Recursion, Backtracking, Greedy, Divide and Conquer and Dynamic Programming. Publisher CareerMonk Publications. First Edition.
[2] Narasimha Karumanchi. Data Structures and Algorithms Made Easy in Java. Publisher CareerMonk Publications. Second Edition.
[3] Hemant Jain. Problem Solving in Data Structures and Algorithms using Java. Publisher Taran Publication.
[4] Goodrich. Tamassia. Goldwasser. Data Structures and Algorithms in Java. International student version. Publisher Wiley Publication. Sixth Edition.
[5] A. A. Puntambekar. Design and Analysis of Algorithm. Publisher Technical Publication. First Edition.
[6] Minal P. Nerkar. A. A. Puntambekar. Data Structure and Algorithm. Publisher Technical Publication.

