ISSN No:-2456-2165

# A Comparative Study between the Greedy and Dynamic Technique for Better Optimization Performance

Mirza Samiulla Beg, Ph.D. Scholar, Department of Computer Science & IT, AKS University,Satna
Akhilesh A. Waoo, Head, Department of Computer Science & IT, AKS University, Satna

**Abstract:- Greedy algorithms can be classified as blind. In a simple way, it always looks to the future and does not look back on the past. All the greed you are trying to achieve is trying to accumulate the best value in the solutions at hand, even though some of the solutions may be very useful in the future. The Dynamic (DP) system is different from greed in the way in which an improved solution is selected. As mentioned earlier, greedy people are constantly on the lookout for profits without regard for the future or the past. DP produces all possible solutions from handmade solutions, and then re-evaluates and re-runs all of them to choose the best solution. In this paper, I want to tell you that a strong process gives better results than the method of greed. It can therefore be said that if we apply the dynamic techniques in the Artificial Bee Colony Algorithm, then it will give us better results. In this paper, I have shown that motivation strategies are better.**

***Keywords:-*** *WSN (Wireless Sensor Network), ABC (Artificial Bee Colony) Algorithm, DP (Dynamic Technique), BS (Base Station), IoT (Internet of Things).*

## I. INTRODUCTION

The greedy algorithm can be a group of algorithms with one common feature, making it the easiest choice in each area for each step without looking at the programs. Therefore, the essence of the greed algorithm can be a task of choice: given the group of options, select the best option currently. Due to the myopic nature of the greedy algorithm, (as expected) is incorrect in several problems. However, certain problems can be easily solved using the greedy algorithm, which can be shown to be correct.

## II. GREEDY ALGORITHM

The greedy algorithm, because the name suggests, always makes choices that seem the easiest now. this suggests that it makes a good choice in the area within the hope that this choice will create the right global solution.Assume that you have a purposeful task that needs to be prepared (or enlarged or reduced) somewhere. The greedy algorithm makes greedy choices at every step to ensure that the targeted work is done. The greedy algorithm has only one gun to calculate the right solution so it doesn't go back and change the selection. Many greedy algorithms are incorrect.

**Advantages and Disadvantages - Greedy Algorithm**

1. It's easy to come back with a greedy algorithm (or even more greedy algorithms) for drag.
2. Analyzing the duration of the operation of greedy algorithms will generally be much easier than other methods (like Divorce and Win). Separating and winning the process, it is not clear whether the process is fast or slow. This is common because at each repetition level the size is smaller and therefore the number of minor problems increases.
3. The hard part is that in the greedy algorithms you must find it very difficult to know the issues of righteousness. Even with the right algorithm, it is difficult to prove why it is correct. Proving that the selfish algorithm is right is more art than science. Includes tons of art.

## III. ARTIFICIAL BEE COLONY (ABC) ALGORITHM

The Artificial Bee Colony (ABC) algorithm is an optimization algorithm that mimics the performance of a bee colony and was first proposed by Karaboga in 2005 to use the actual parameter.In this mathematical example, our bee colony is made up of three types of bees: Worker Bees, which will work in collecting food in the hive at a specific food source. Bees are called Onlooker Bees, which will guard workers to ensure that a particular food source is no longer suitable, as well as Scout bees, which will seek out new sources of food.

In the ABC algorithm, the food source is defined as a position in the search space (solution for the problem of efficiency), and initially, the number of feed sources is equal to the number of bees in the hive. The quality of the food source is determined by the amount of objective activity in that position (the amount of consistency).The emerging behavior of bees can be summed up in a few steps:

- Bees start randomly exploring the environment in search of good food sources (value).
- After finding a source of food, the bee becomes a worker bee and begins to extract food from the discovered source.
- The worker bee returns to the hive with nectar and releases the nectar. After extracting the nectar, he can return directly to his found domain directly or share information about his source location by performing a dance on the dance floor.

- When the food supply is exhausted, the worker bee becomes a spout and begins randomly in search of a new source of food.
- Observing bees waiting in the hive observe bees working in the collection of their food source and select the source among the most profitable sources.
- The choice of food source is equal to the quality of the source (number of solids).

To do these things we can use the method of greed and use Dynamic. So here is the definition of both algorithms. And make it clear that Dynamic is better than stubborn.

## IV. DYNAMIC TECHNIQUE

Dynamic planning can be a way to divide issues into smaller issues and save results for future purposes so that we don't have to re-calculate the results. The underlying problems are designed to add a common solution is understood as a good site for reconstruction. Excessive use of a powerful system to open up performance problems. Here, performance issues mean that if we are trying to find a minimum or a great drag solution. A powerful system ensures finding the right solution to drag when the answer is available. The definition of a powerful system states that it is a way to solve a posh problem by first logging into a set of simple problems, solving each problem at once, and then keeping their solutions to avoid duplicate calculations.

## V. REVIEW OF LITERATURE

K. Lin et al [1] presented a remote sensor organization (WSN) ordinarily works in a problematic remote climate with energy limitations. Numerous specialists are fundamentally intrigued by the energy mindfulness and correspondence dependability of WSNs to amplify network lifetime. Be that as it may, managing the clashing issues of further developing energy proficiency and adaptation to internal failure all the while is a difficult undertaking. Most past investigations have shown that the two issues can be drawn nearer by utilizing either information connection or organization layer conventions. They present a cross-layer convention, which incorporates a multipath steering convention and an information interleaving strategy dependent on the Reed-Solomon code. They define the issue of choosing sensor transmission ways as a backpack issue and tackle it by an avaricious calculation. Our multipath directing convention then, at that point, empowers every sensor to choose numerous transmission ways utilizing the proposed streamlining calculation. Based on numerous transmission ways, the method of information interleaving is utilized by utilizing the Reed-Solomon code to give solid information transmission. Reenactment results show that our plan beats the current multipath directing conventions concerning the organization's lifetime since it adjusts energy utilization and advances correspondence unwavering quality.

X. Wang et al [2] proposed it is a generally expected plan in the majority of the current geography plans of remote sensor network that the bunch head hubs can speak with a base station (BS) hub straightforwardly, which causes that the group head hubs burn-through an excessive amount of energy and become the bottleneck of organization execution. Accordingly, a WSN geography calculation dependent on avaricious most limited ways is proposed in this paper. Right off the bat, choosing the spine hubs of the organization and building the spine organization to send messages, and permitting the spine hubs and group heads to exist independently. Also, the spine hubs construct a solitary source briefest way directing table dependent on the covetous calculation and select the most appropriate most limited way to send information as per the utilization of the spine hubs. Thirdly, choosing the group heads is dependent on the LEACH convention, in which the bunch heads are answerable for gathering and packing the information, choosing the closest spine hub to send the information. It is shown by information estimation and investigation that the proposed geography conspires in this paper have a more steady transmission network structure, less weight, and less remaking recurrence, and the group heads are just answerable for moving the prepared information to the spine organization. Also, the bunch head utilization and the general organization utilization are altogether worked on contrasted and LEACH.

Seyed Reza Nabavi et al [3] suggested due to the widespread use of communication networks and the ease of transmitting and collecting information about these networks, WSN wireless networks are becoming increasingly popular. The usability of any area without the need for environmental monitoring and engineering of these networks has led to its increasing use in various fields. Moving data from a sensor node to a sink, so that the power of the node is consumed uniformly and network life can be reduced, is one of the most important challenges for wireless sensor networks. Most wireless networks do not have the infrastructure, and embedded sensor nodes have limited power. Therefore, the initial term of the wireless node power based on network messaging may disrupt the entire network process. In this paper, the object is designed to determine the correct path to WSN based on the multi-objective greed method of the nearest route. The proposed model is presented in this way to transfer sensor network information to the base station of the desired applications. In this way, the sensor nodes are identified as adjacent nodes based on their distance. The power of all places is initially almost equal, diminished by the transfer of information between places. This way, when a node hears a message, it looks for several data transfer objects in its nearby nodes and selects a node with several larger objects such as the next hop. Imitation results show that power consumption on network grids is almost equally presented, and network life is reduced by a small slope that provides optimal power consumption to networks. Also, the packet transfer delay in the network is up to 450 milliseconds of data transmission between 15 nodes and 650 connections. Besides, network penetration increases by about 97%. It also shows better performance compared to other previous methods in terms of testing.

Ahmed RedhaMahlous et al [4] presented they have seen an expanding interest in creating and planning Wireless Sensor Networks (WSNs). WSNs comprise an enormous number of hubs, with remote correspondences and calculation capacities that can be utilized in an assortment of spaces. It has been utilized in regions that have direct contact with observing and assembling information, to name not many, wellbeing checking, military observation, land checking (Earthquakes, Volcanoes, Tsunami), agribusiness control, and some more. Be that as it may, the plan and execution of WSNs face many difficulties, because of the force limit of sensor hubs, arrangement, and confinement, information directing and information collection, information security, restricted data transmission, stockpiling limit, and organization of the board. It is realized that Operation Research (OR) has been broadly utilized in various regions to take care of advancement issues, for example, further developing organization execution and amplifying the lifetime of framework. They present the latest OR-based procedures applied to take care of various WSNs issues: the hub booking issue, energy the executive's issues, hubs apportioning issues, and other WSNs related complex issues. Distinctive Operational Research methods are introduced and examined in subtleties here, including chart hypothesis-based procedures, direct programming, and blended whole number programming-related methodologies.

Sadek et al [5] proposed one of the focal correspondence frameworks of the Internet of Things (IoT) is the IEEE 802.15.4 norm, which characterizes Low Rate Wireless Personal Area Networks (LR-WPAN). To share the medium genuinely in a non-signal empowered mode, the standard uses Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). The idea of associated objects concerning different asset limitations makes them powerless against digital assaults. Quite possibly the most forceful Do assaults is the voracious conduct assault which intends to deny genuine hubs to admittance to the correspondence medium. The covetous or egotistical hub might disregard the appropriate utilization of the CSMA/CA convention, by altering its boundaries, to take however much data transmission as could be expected on the organization, and afterward consume admittance to the medium by denying real hubs of correspondence. In light of the examination of the contrast between boundaries of ravenous and authentic hubs, they propose a technique dependent on the limit system to distinguish avaricious hubs. The recreation results show that the proposed component gives a recognition proficiency of 99.5%.

J. P. Mohanty et al [6] presented in the specially appointed remote organization, there is no predefined framework. In this way, hubs speak with one another through peer interchanges. For powerful correspondence, an associated overwhelming set (CDS) can be utilized as a virtual spine for the organization. Notwithstanding, building a base-associated ruling set is an NP-Complete issue. In the writing, numerous estimation calculations have been accounted for. In this paper, they propose a conveyed three-stage insatiable guess calculation. In our calculation, the hubs just store one-jump neighborhood data to track down the following dominators. They likewise propose an approach to lessen the CDS size by minimizing a portion of the current dominators after the development of CDS. The reproduction result shows that our CDS development conspire beats every one of the current CDS development calculations as far as CDS size for arbitrarily dispersed hubs. Our calculation holds the presentation proportion of (4.8 + ln5)opt + 1.2 and time intricacy of O(D), where pick being the size of the ideal CDS and D is the measurement of the organization.

Q. Q. Shi et al [7] suggested geographic directing conventions for remote sensor organizations (WSNs) have gotten more consideration lately and ravenous sending calculation is the primary part of geographic steering. They research the sending models in ravenous sending calculations and present an eager steering calculation utilizing another basis joining the qualities of both distance-based rule and course-based rule. Reenactment is given to contrast the exhibition of our calculation and those of the calculation with the distance-based model and the calculation with the heading-based standard. The outcomes show that our proposed calculation is a favored choice as far as the compromise between change postponement and energy utilization in the steering.

Y. Xin et al [8] presented in the utilization of remote sensor organizations (WSN), the equilibrium of energy utilization assumes a significant part in broadening the existing pattern of WSN. Focus on the energy utilization of remote sensors organization, a powerful group put together directing convention-based for the avaricious calculation (GDP). In the convention, hubs run for the group head as per the energy and area. The chosen group head has an ideal worth of energy and area. At the point when the group is framed, the sink hub starts the solicitation of directing foundation; the bunch heads pick the hubs that send the bundle as the upper hubs as per energy and bounces build-up to the sink hub. After a time of the organization working, then, at that point, pick hubs as indicated by the energy worth and distance as the bunch head indeed to stay away from network disappointment as a result of one hub disappointment. The convention adjusts the energy utilization of the organization's impact and expands the existing pattern of the whole organization.

S. Bousnina et al [9] proposed Virtual Sensor Networks (VSNs) imagine the making of universally useful remote sensor networks which can be effortlessly adjusted and arranged to help multifold applications with heterogeneous prerequisites, interestingly, with the traditional methodology of remote sensor networks upward enhanced on one explicit undertaking/administration. The actual heart of VSNs' vision is the capacity to powerfully distribute shared actual assets (handling power, data transmission, stockpiling) to various approaching applications. In this unique circumstance, they tackle the issue of ideally dispensing shared assets in VSNs by proposing a proficient avaricious heuristic that intends to expand the complete income out of the sending of different simultaneous applications while thinking about the intrinsic

limits of the common actual assets. The proposed heuristic is tried on reasonable organization occasions with remarkable exhibitions as far as execution time while keeping the hole as for the ideal arrangement restricted (beneath 5% in the tried conditions).

L. Ban-teng et al [10] suggested Remote Sensor Networks(WSN) are a problem area of the examination of remote organizations right now, the key to accomplishing a proficient transmission business is to control hub energy and further develop the organization lifetime in remote sensor organizations. The paper first uses Boolean detecting model dependent on Poisson direct interaction toward recognize the capacity of the pace of inclusion and the hub thickness in-unit region and afterward computes the all outnumber of hubs in the locale, next utilize the avaricious technique of the Prim calculation to discover a traversing tree with the most extreme weight, and builds a surmised answer for the base associated ruling set. To control the upheavals of the hubs, make the hubs in the crossing tree work, and different hubs are in a rest state. Finally, further examination of the connection between the number of hubs in associated overwhelming and the inclusion sweep.

H. A. Hamzah et al [11] presented accuracy cultivating as a cultivating the board idea dependent on noticing, estimating, and reacting to the shifting harvests' necessities. For this examination, Wireless Sensor Network (WSN) is proposed to be carried out inaccuracy cultivating to go about as a choice emotionally supportive network for the ranchers. Notwithstanding, WSN customary framework specifically the Direct Transmission (DT) convention experiences high energy utilization, decreasing the observing capacity of the ranchers because of the quick energy-exhausting sensor hubs. Recognizing this issue, analysts had created numerous conventions, for example, Quality of Service (QoS), Low Energy Adaptive Clustering Hierarchy (LEACH), Location-based, and Power-Efficient Gathering in Sensor Information System (PEGASIS). For this exploration, PEGASIS is chosen for its high energy proficiency and similarity with the proposed framework. As distances influence significantly energy utilization, Particle Swarm Optimization (PSO) is created to supplant ravenous calculation in PEGASIS to diminish the distances of information transmission. From the tests, PSO can diminish the absolute chain distance (TCD) by dependent upon 7.69% in contrast with the insatiable calculation.

Djahel et al [12] proposed while the issue of ravenous conduct at the MAC layer has been broadly investigated with regards to the remote neighborhood, its review for multi-bounce remote organizations is still just about a neglected and unexplained issue. Without a doubt, in a remote neighborhood, a passageway for the most part advances bundles sent by remote hubs over the wired connection. For this situation, a voracious hub can undoubtedly get more transmission capacity share and starve any remaining related battling hubs by astutely controlling the MAC layer boundaries. In any case, in a remote specially appointed climate, all bundles are communicated in a multi-bounce style over remote connections. Consequently, if a ravenous hub acts correspondingly as in the WLAN case, attempting to starve its neighbors, then, at that point, its next bounce sending hub will likewise be forestalled to advance its traffic, which prompts a start to finish throughput breakdown. In this paper, they show that to have a more helpful insatiable conduct in remote specially appointed organizations, a hub should embrace an unexpected methodology in comparison to WLAN to accomplish its very own superior exhibition streams. They present a technique to dispatch a particularly insatiable assault in a proactive directing based remote specially appointed organization. Through the broad reproductions, the acquired outcomes show that by applying the proposed calculation, a ravenous hub can acquire more data transmission than its neighbors and keep the start to finish throughput of its streams profoundly sensible.

Tzu-Chiang et al [13] suggested with the benefit of remote organization innovation, there are assortments of portable applications which make the issue of remote sensor networks a well-known exploration region lately. As the remote sensor network hubs move self-assertively with the geography quick-change highlight, portable hubs are frequently defied with the void issue which will start parcel losing, retransmitting, rerouting, extra transmission cost, and force utilization. When sending parcels, they would not foresee void issues happening ahead of time. Consequently, how to work on geographic steering with void evasion in remote organizations turns into a significant issue. In this paper, they proposed a covetous topographical void steering calculation to take care of the void issue for remote sensor organizations. They utilize the data of the source hub and void region to attract two digressions to shape a fan scope of the current void which can declare void keeping away from the message. Then, at that point, they use source and objective hubs to define a boundary with a point of the fan reach to choose the following sending neighbor hub for directing. In a unique remote sensor network climate, the proposed ravenous void staying away from calculation can be additional efficient and more proficient to advance bundles and work on the current geological void issue of remote sensor organizations.

## VI. IMPLEMENTATION OF GREEDY ALGORITHM

Greedy algorithms can be classified as blind. In its simplest way, it looks always for the future and doesn't look back to the past. All that Greedy is trying to achieve is to try to collect the best benefit from the solutions in hand, regardless of whether some other solutions can be more beneficial at another moment of time in the future.

```
# Greedy Algorithm for a Optimisation Problem
# Defined a class for item,with its name, value, and cost
class Itm(object):
  def __init__(self, name, val, cost):
    self.name = name
self.val = val
self.cost = cost
  def getvalue(self):
```

```
    return self.val
  def getcost(self):
    return self.cost
  def __str__(self):
    return self.name

# Defining a function for building a List
# which generates list of items that are
# available at supermart
def buildlist(names, values, costs):
  menu = []
  for i in range(len(names)):
menu.append(Itm(names[i], values[i], costs[i]))
  return menu

# Implementation of greedy algorithmto choose one of the optimum choice
def greedy (items, maxcal, keyfunction):
itemscopy = sorted(items, key = keyfunction, reverse = True)
  result = []
totalval = 0
totalcal = 0

  for i in range(len(items)):
    if (totalcal + itemscopy[i].getcost() <= maxcal):
result.append(itemscopy[i])
totalval = totalval + itemscopy[i].getvalue()
totalcal = totalcal + itemscopy[i].getcost()
  return (result, totalval,totalcal)

# Main Function# All values are random
names = ['Ball', 'Gloves', 'Notebook', 'Bagpack', 'Charger', 'Pillow', 'Cakes', 'Pencil']
values = [89,90,95,100,90,79,50,10]
costs = [123,154,25,145,365,150,95,195]

Itemrs = buildlist(names, values, costs)
maxcost = 500 # maximum money he have to spend
taken, totvalue,totalcal = greedy(Itemrs, maxcost, Itm.getvalue)
print('Total value taken : ', totvalue,totalcal)
# Printing the list of item slected for optimum value
for i in range(len(taken)):
  print('  ', taken[i])
```

**Output:** Total value taken : 374 , 447

**Dynamic-Programming Algorithm**

Dynamic programming (DP) is different from greedy in the way in which the optimized solution is selected. As mentioned earlier, greediness always seeks the maximum available profit without looking for the future or the past. DP generates all feasible solutions from the solutions in hand, then iterates again through all of them to select the best solution.

```
class Item:
  def __init__(self,weight,value):
self.weight = weight
self.value = value
  def getWeight(self):
    return self.weight
  def getValue(self):
    return self.value
  def __str__(self):
    return str(self.weight)+" , "+ str(self.value)

  def getData(items):
    w = []
    v = []
    for ob in items:
w.append(ob.weight)
v.append(ob.value)
    return (w,v)

def buildItem(W,V,keyfunction,reverse):
  items = []
  for i in range(len(W)):
ob = Item(W[i],V[i])
items.append(ob)
  return items

def dynamic(maxWeight,items):
weight,value = Item.getData(items)
  n = len(value)
  S = [[0 for x in range(maxWeight+1)] for k in range(n+1)]
  for x in range(1, maxWeight+1):
    for k in range(1, n+1):
      S[k][x] = S[k-1][x]
      if weight[k-1] <= x andS[k-1][x-weight[k-1]]+value[k-1]>S[k][x]:
          S[k][x] = S[k-1][x-weight[k-1]] + value[k-1]
  data = list()
  for i in range(len(S)-1,-1,-1):
arr = S[i]
maxNum = max(arr)
    #print("\n>>> M >> ",maxNum)
    if maxNum not in S[i-1] and maxWeight> weight[i-1]:
data.append((weight[i-1],value[i-1]))
      #print(data)
maxWeight-= weight[i-1]
      if maxWeight==0:
        break
  return data
maxWeight = 500
w = [100,90,79,89,90,95,50,10]
v = [145,365,150,123,154,25,95,195]

itemList1 = buildItem(w,v,Item.getWeight,False)
result = dynamic(maxWeight,itemList1)
print(result)
wSum = 0
vSum = 0
for it in result:
wSum += it[0]
vSum += it[1]
print("Total Weight : " , wSum," Price : ",vSum)
```

**Output:**[(145, 100), (25, 95), (154, 90), (123, 89)]
Total Weight: 447 Price: 374

**Test Cases for Dynamic vs Greedy**

**Test 1 : maxcost = 500 # maximum spending**
values = [89,90,95,100,90,79,50,10]
costs = [123,154,25,145,365,150,95,195]
According Greedy:Weight:374 , Price : 447
According Dynamic:Weight:447,Price: 374

In this test find the maximum weight and minimum price. The greedy test gives the weight 374 and price 447 which is not an accurate result. When implement with the dynamic test then get the weight 447 and price is 374 which is an accurate result. So the dynamic test gives the maximum weight and minimum price.

**Test 2:maxcost = 600 # maximum spending**
values = [89,90,95,100,90,79,50,10]
costs = [123,154,25,145,365,150,95,195]
According Greedy:Weight:453,Price: 597
According Dynamic:Weight:597,Price: 453

In this test find the maximum weight and minimum price. The greedy test gives the weight 453 and price 597 which is not an accurate result. When implement with the dynamic test then get the weight 597 and price is 453 which is an accurate result. So the dynamic test gives the maximum weight and minimum price.

**Test 3:maxcost = 400 # maximum spending**
values = [10,20,15,12,14,10,12,16,14,18,12,12,10,12,23]
costs = [67,78,45,34,12,78,56,45,23,12,45,67,89,34,32]
According Greedy:Weight:99,Price: 337
According Dynamic:Weight:143,Price: 371

In this test find the maximum weight and maximum price. The greedy test gives the weight 99 and price 337 which is not an accurate result. When implement with the dynamic test then get the weight 143 and price is 371 which is an accurate result. So the dynamic test gives the maximum weight and minimum price.

**Minimum and Maximum Path Using Greedy and Dynamic Technique**

**Node.py**

```
class Node:
  def __init__(self,name):
    self.name = name
self.next = None
  def setConnectedNodes(self,lst):
    if type(lst) is list:
self.next = lst
    else:
      raise Exception("Node list is required !")
  def getName(self):
    return self.name
  def getConnectedNodes(self):
    return self.next
```

**Greedy.py**

```
from node import Node
def findMax(wl):
  #print(">>> ",wl)
  key = None
  weight = None
  for data in wl:
    if key is None:
      key = data.get('key')
      weight = data.get('weight')
    else:
      if data.get('weight') > weight:
        key = data.get('key')
        weight = data.get('weight')
  return {'key':key , 'weight' : weight}

def findMin(wl):
  #print(">>> ",wl)
  key = None
  weight = None
  for data in wl:
    if key is None:
      key = data.get('key')
      weight = data.get('weight')
    else:
      if data.get('weight') < weight:
        key = data.get('key')
        weight = data.get('weight')
  return {'key':key , 'weight' : weight}

def greedy(source,destination,isMinimum=True):
  path = {'key': source.getName() , 'weight' : 0}
nd = source
  while True:
    if nd.getConnectedNodes() is None and nd.getName() is
not destination.getName():
      raise Exception("Destination Node Not Found !")
elifnd.getConnectedNodes() is None and nd.getName() is
destination.getName():
      return path
    else:
      if len(nd.getConnectedNodes())==1:
        node = nd.getConnectedNodes()[0]

newkey = path.get('key') + node.get('node').getName()
newweight = path.get('weight') + node.get('weight')
path.update({'key':newkey,'weight':newweight})
nd = node.get('node')
      else:
weightlst = []
        for nodelist in nd.getConnectedNodes():
          node = nodelist.get('node')
          weight = nodelist.get('weight')
weightlst.append({'key':node,'weight':weight})

        if isMinimum:
          final = findMin(weightlst)
        else:
          final = findMax(weightlst)

newkey = path.get('key')+ final.get('key').getName()
newweight = path.get('weight')+ final.get('weight')
path.update({'key':newkey,'weight':newweight})
nd = final.get('key')
```

ISSN No:-2456-2165

**Dynamic.py**

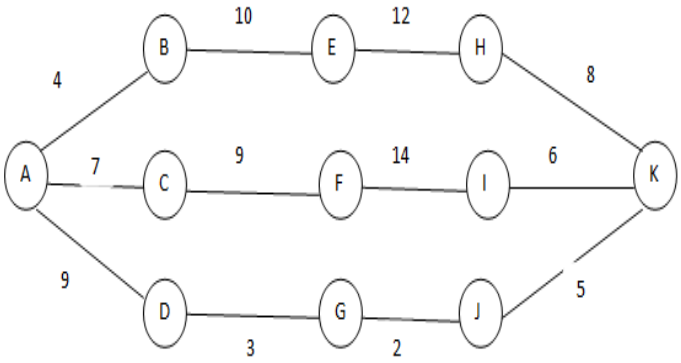
```
from node import Node
def findMax(wl):
  #print(">>> ",wl)
  key = None
  weight = None
  for data in wl:
    if key is None:
      key = data.get('key')
      weight = data.get('weight')
    else:
      if data.get('weight') > weight:
        key = data.get('key')
        weight = data.get('weight')
  return {'key':key , 'weight' : weight}

def findMin(wl):
  #print(">>> ",wl)
  key = None
  weight = None
  for data in wl:
    if key is None:
      key = data.get('key')
      weight = data.get('weight')
    else:
      if data.get('weight') < weight:
        key = data.get('key')
        weight = data.get('weight')
  return {'key':key , 'weight' : weight}

def dynamic(source,destination,isMinimum=True):
  path = []
nd = source
  while True:
    if nd.getConnectedNodes() is None and nd.getName() is
not destination.getName():
      raise Exception("Destination Node Not Found !")
elifnd.getConnectedNodes() is None and nd.getName() is
destination.getName():
      return {'key':nd.getName(),'weight':0}
    else:
weightlst = []
      for nodelist in nd.getConnectedNodes():
        node = nodelist.get('node')
        nm = dynamic(node,destination)
        #print(nm)

        name = nd.getName() + nm.get('key')
        weight = nodelist.get('weight') + nm.get('weight')
weightlst.append({'key':name,'weight':weight})
      if len(nd.getConnectedNodes())==1:
        final = weightlst[0]
      else:
        if isMinimum:
          final = findMin(weightlst)
        else:
          final = findMax(weightlst)
        #print(source.getName() , " final :  " , final)
      return final
```

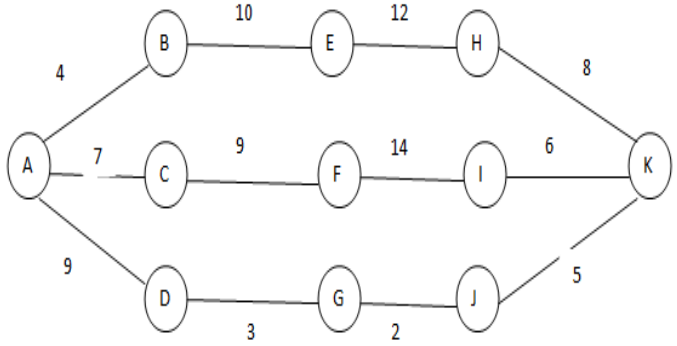
**Find Minimum Path Using Greedy and Dynamic Technique**




Greedy Result: {'weight': 34, 'key': 'ABEHK'}
Dynamic Result: {'key': 'ADGJK', 'weight': 19}

In this graph find the minimum path using the greedy and dynamic techniques. When finding the minimum path using the greedy technique get that path 'ABEHK' and weight is 34. When finding the path using the dynamic technique get that path 'ADGJK' and weight is 19. So here the dynamic technique gives the minimum path which is 19. So the dynamic technique is best as compared to the greedy technique.

**Find the Maximum path using Greedy and Dynamic Technique.**




Greedy Result:{'key': 'ADGJK', 'weight': 19}
Dynamic Result: {'weight': 36, 'key': 'ACFIK'}

In this graph find the maximum path using the greedy and dynamic techniques. When finding the maximum path using the greedy technique get that path 'ADGJK' and weight is 19. When finding the path using the dynamic technique get that path 'ACFIK' and weight is 36. So here the dynamic technique gives the maximum path which is 36. So the dynamic technique is best as compared to the greedy technique.

## VII. CONCLUSION

In this paper, a comparative study of the Greedy and Dynamic techniques has been done. This study is done by taking a graph. On which who uses the short and long path to go from the source to the destination. When we studied then we came to know that Dynamic Technique is better

than Greedy Technique, because when we search short path then Greedy Technique using this 'ABEHK' path gives us 34 as result, while Dynamic Technique Using this 'ADGJK' path gives us 19 as the result, which is the best result. Similarly, when we search for longer paths, the Greedy technique gives us 19 as a result of using this 'ADGJK' path, while the Dynamic technique gives us 36 as a result of using this 'ACFIK' path. This is a long path. In this way, we can say that the dynamic technique is best than the greedy technology.

## REFERENCES

[1]. K. Lin, P. Wang and T. Hong, "A greedy algorithm in WSNs for maximum network lifetime and communication reliability," 2015 IEEE 12th International Conference on Networking, Sensing and Control, 2015, pp. 87-92, doi: 10.1109/ICNSC.2015.7116015.

[2]. X. Wang, J. Zhao, J. Hou, X. Tang, H. Wu and P. He, "Research on WSN Topology Algorithm Based on Greedy Shortest Paths," 2020 39th Chinese Control Conference (CCC), 2020, pp. 5215-5220, doi: 10.23919/CCC50068.2020.9189355.

[3]. Seyed Reza Nabavi, NafisehOsatiEraghi, Javad Akbari Torkestani, "WSN Routing Protocol Using a Multiobjective Greedy Approach", Wireless Communications and Mobile Computing, vol. 2021, Article D 6664669, 12 pages, 2021. https://doi.org/10.1155/2021/6664669

[4]. "Operation Research Based Techniques in Wireless Sensors Networks" written by Ahmed RedhaMahlous, Mohamed Tounsi, published by Communications and Network, Vol.9 No.1, 2017

[5]. Sadek, F.S.; Belkadi, K.; Abouaissa, A.; Lorenz, P. Identifying Misbehaving Greedy Nodes in IoT Networks. Sensors 2021, 21, 5127. https://doi.org/10.3390/s21155127

[6]. J. P. Mohanty and C. Mandal, "A distributed greedy algorithm for construction of minimum connected dominating set in wireless sensor network," 2014 Applications and Innovations in Mobile Computing (AIMoC), 2014, pp. 104-110, doi: 10.1109/AIMOC.2014.6785527.

[7]. Q. Q. Shi, H. Huo, T. Fang and D. R. Li, "A Modified Greedy Distance Routing Algorithm for Wireless Sensor Networks," 2008 China-Japan Joint Microwave Conference, 2008, pp. 197-200, doi: 10.1109/CJMW.2008.4772405.

[8]. Y. Xin, X. Guang-hua and C. Xiao-jun, "The Research on Routing Protocol of Sense Wireless Network Based on the Greedy Algorithm," 2009 International Conference on Networks Security, Wireless Communications and Trusted Computing, 2009, pp. 558-561, doi: 10.1109/NSWCTC.2009.159.

[9]. S. Bousnina, M. Cesana, J. Ortín, C. Delgado, J. R. Gállego and M. Canales, "A greedy approach for resource allocation in Virtual Sensor Networks," 2017 Wireless Days, 2017, pp. 15-20, doi: 10.1109/WD.2017.7918108.

[10]. L. Ban-teng, C. You-rong, Z. Kai and J. Hua, "The Research of Wireless Sensor Networks Optimization Algorithm Based on the Energy Control," 2010 Third International Symposium on Information Processing, 2010, pp. 420-422, doi: 10.1109/ISIP.2010.83.

[11]. H. A. Hamzah, N. Tuah, K. G. Lim, M. K. Tan, L. Zhu and K. T. K. Teo, "Data Transmission in Wireless Sensor Network with Greedy Function and Particle Swarm Optimization," 2019 IEEE 7th Conference on Systems, Process and Control (ICSPC), 2019, pp. 172-177, doi: 10.1109/ICSPC47137.2019.9068052.

[12]. Djahel, Soufiene&Naït-Abdesselam, Farid & Turgut, Damla. (2010). An Effective Strategy for Greedy Behavior in Wireless Ad hoc Networks. 1 - 6. 10.1109/GLOCOM.2009.5425412.

[13]. Tzu-Chiang, C., Jia-Lin, C., Yue-Fu, T. , Sha-Pai, L.. "Greedy Geographical Void Routing for Wireless Sensor Networks". World Academy of Science, Engineering and Technology, Open Science Index 78, International Journal of Computer and Information Engineering (2013), 7(6), 769 - 777.