4.1 INTRODUCTION

In this chapter, Dynamic programming model has been developed to produce different sets of traffic matrices as stage wise for finding of the shortest path from each source to different destinations. The proposed heuristics are developed based on HLDA and implemented on the traffic matrix generated through Dynamic Programming are (1) Dynamic Programming Heuristic Approach (DP-H) (2) Dynamic Programming Source based Approach (DP-S) and (3) Dynamic Programming Destination Approach (DP-D) is implemented to find the shortest path for formation of Virtual Topology A performance measure is made on stage wise 4-Node, 14-Node NSFNET traffic matrices. The other way of arbitrary chosen Dynamic traffic matrix generated randomly at different stages is used for formation of Virtual Topology. The proposed heuristics are developed based on HLDA implemented on Dynamic traffic matrix are named as (1) Dynamic Value Heuristic Approach (DVH), (2) Dynamic Value Source Based Approach (DVS), and (3) Dynamic Value Destination Based Approach (DVD).

The cost of the link connectivity nodes in a network which is treated as directed network represented as a traffic matrix. The implementation of Dynamic programming is used to find each source to different destination. Initial network Constructed referred to as stage-1 will be extended to stage-2. Stage-2 means finding of intermediate node with least cost we will update the network is referred to as stage-2 continued up to full-fledged least cost traffic matrix (stage-5).

Representation of the network based on the directed network and as per the existing traffic matrix the following characteristics are involved

- **Stages:** The given traffic matrix based on the topology the length of every node will be divided into range of sub problems, referred as stages.
- **Decision:** In each stage, there may be multiple selections out of that the most effective is shortest path (the least cost).
- **State:** A state indicates that the variables in stage area unit referred to as state
variable.

- **Policy**: A policy could be a rule that determines the choice at every stage. A policy is named associate optimum policy if it's globally optimum.

- **Principle of Optimality**: An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.

The use of dynamic programming approach is for finding the shortest path between any two given two nodes. The input for the algorithm in a network $G = \langle V, E \rangle$, and a set of weights $w_{i,j}$ for each $(i, j) \in E$. The algorithm finds the shortest simple path from node $s$ to node $t$. Input for the algorithm is an adjacency matrix of the graph $G$. The cost $(i, j)$ is computed as follows initially, the matrix $D(0)$ is initialized as cost $(i, j)$. Then, in subsequent iterations, the next matrix $D$ is computed. Now $D(1)$ is computed with only one intermediate node $\{1\}$. $D(2)$ is computed with only two intermediate nodes $\{1, 2\}$. In general, $D(k)$ is computed from $D(k-1)$ and there are only two possibilities for computing it, which are as follows:

- The first possibility is that node $k$ is not involved. Since the node is not part of the path, only nodes $\{1, 2, \ldots, k-1\}$ are considered. Hence, $D_{ij}(k) = D_{ij}(k-1)$.

- The second possibility is that node $k$ is used as an intermediate node. In this case, one can split the path as $I \rightarrow k$ and $k+1 \rightarrow j$. If the path $I \rightarrow j$ is the shortest path, then the paths $I \rightarrow k$ and $k + 1 \rightarrow j$ should be the shortest paths as well.

The shortest path between $i$ and $j$ is computed as follows

$$D_{ij}(k) = \min \{D_{ij}(k-1), D_{ik}(k-1) + D_{kj}(k-1)\}$$

The algorithm proceeds subsequently as $D(0), D(1), D(2)$, and $D(n)$. It can be observed that $D(n)$ entries represent the shortest path between any pairs of vertices.

**Step 1**: Read weighted graph $G = \langle V, E \rangle$.

**Step 2**: Initialize $D[i, j]$ as follows
Cost(i, j) = \begin{cases} 
0 & \text{if } i=i, 1 \leq i \leq n \\
W_{ij} & \text{if } <i, j> \in E(G) \\
\infty & \text{if } <i, j> \notin E(G) 
\end{cases}

**Input:** A weighted graph, represented by its weights matrix

**Output:** produces the matrix distances [ ] [ ] of values representing the all pair shortest path

All pair shortest path (G)

For each u ∈ V do

For each v ∈ V do

Dist[u][v] = ∞

Pred[u][v] = 1

Dist[u][u] = 0

For each neighbour v of u do

Dist[u][v] = weight of edge (u, v)

Pred[u][v] = u

For each t ∈ v do

For each u ∈ v do

For each v ∈ v do

New Len = dist[u][t] + dist[t][v]

If (new Len < dist[u][v]) then

Dist[u][v] = new Len

Pred[u][v] = pred[t][v]

End
4.2 Implementation of Dynamic programming on 4-Node in all pair the shortest path

Fig: 4.1: 4-Node Directed network

Fig 4.1: 4-Node network these networks to find out the shortest path from each source to different destination using Dynamic programing. Link connectivity networks in direct network cost are initialized randomly and represented in traffic matrix.

\[
\begin{bmatrix}
0 & 8 & \infty & 1 \\
\infty & 0 & 1 & \infty \\
4 & \infty & 0 & \infty \\
\infty & 2 & 9 & 0
\end{bmatrix} \quad \text{Stage-1}
\]

4-Node Traffic matrix

In the above 4-Node traffic matrix, directed graph is initialized randomly and represented in a traffic matrix. It will check neighbors on supported physical network, referred to as stage-1. That means intermediate node is zero

\[
\begin{bmatrix}
0 & 8 & \infty & 1 \\
\infty & 0 & 1 & \infty \\
4 & 12 & 0 & 5 \\
\infty & 2 & 9 & 0
\end{bmatrix} \quad \text{Stage-2}
\]
In the above 4-Node traffic matrix one intermediate node is added in all directions forms the shortest path referred as stage-2.

\[
\begin{bmatrix}
0 & 8 & 9 & 1 \\
\infty & 0 & 1 & \infty \\
4 & 12 & 0 & 5 \\
\infty & 2 & 3 & 0
\end{bmatrix}
\]
Stage- 3

In the above 4-Node traffic matrix two intermediate nodes is added in all directions forms the shortest path referred as stage-3. This values are current best (least-cost) values.

\[
\begin{bmatrix}
0 & 3 & 4 & 1 \\
5 & 0 & 1 & 6 \\
4 & 12 & 0 & 5 \\
13 & 2 & 3 & 0
\end{bmatrix}
\]
Stage -4

In the above 4-Node traffic matrix three intermediate nodes is added in all directions forms the shortest path referred as stage-3. This values are current best (least-cost) values.

\[
\begin{bmatrix}
0 & 3 & 4 & 1 \\
5 & 0 & 1 & 6 \\
4 & 7 & 0 & 5 \\
7 & 2 & 3 & 0
\end{bmatrix}
\]
Stage-5

In the above 4-Node traffic matrix four intermediate nodes is added in all directions forms the shortest path referred as stage-3. These values are shortest for all direction complete (least-cost) values.

In stage-1 to stage-5 formation of virtual topology, using Dynamic programming Heuristic Approaches in the 4-Node traffic matrices graphically represented in fig 4.2
Fig: 4.1 is a physical network with initial traffic matrix treated as stage-1 on which Dynamic Programming heuristic (DP-H) of Dynamic programming approach is applied. The formation of stage wise 4-Node virtual topology is represented in Fig-4.2. Formation of virtual topologies may took place from stage-1 to stage-5 traffic matrices.

On the same physical network the other heuristics namely Dynamic programming Source based heuristic (DP-S) and Dynamic programming Destination based heuristic (DP-D) of Dynamic programming approach are implemented for formation of Virtual Topology.
The following tables show the results obtained for several objective functions like the utilization of Light paths, Wave Lengths, total number of Physical hops, Hop Weight, Total Hop weight, Average hop Weight, Maximum Congestion and Minimum Congestion at each stage after implementation of the proposed heuristics known as DP-H, DP-S and DP-D of Dynamic Programming approach on 4-Node traffic matrix used up to 2 transceivers.

<table>
<thead>
<tr>
<th>DP-H</th>
<th>Node</th>
<th>Trans</th>
<th>Light</th>
<th>Wave Len</th>
<th>Phy Hop</th>
<th>Hop Wei</th>
<th>Tot Hop</th>
<th>Avg Wei</th>
<th>Max Congestion</th>
<th>Min Congestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>stage1</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>21</td>
<td>21</td>
<td>1</td>
<td>3 &gt; 2 (9)</td>
<td>2 &gt; 0 (4)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>6</td>
<td>34</td>
<td>34</td>
<td>1</td>
<td>0 &gt; 1 (16)</td>
<td>1 &gt; 2 (1)</td>
</tr>
<tr>
<td>stage2</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>22</td>
<td>46</td>
<td>2.091</td>
<td>3 &gt; 2 (9)</td>
<td>2 &gt; 0 (7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>9</td>
<td>36</td>
<td>65</td>
<td>1.806</td>
<td>0 &gt; 3 (13)</td>
<td>1 &gt; 2 (1)</td>
</tr>
<tr>
<td>stage3</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>6</td>
<td>21</td>
<td>63</td>
<td>3</td>
<td>0 &gt; 3 (11)</td>
<td>1 &gt; 2 (4)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>10</td>
<td>37</td>
<td>84</td>
<td>2.27</td>
<td>0 &gt; 3 (16)</td>
<td>3 &gt; 2 (3)</td>
</tr>
<tr>
<td>stage4</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>10</td>
<td>35</td>
<td>89</td>
<td>2.543</td>
<td>2 &gt; 0 (20)</td>
<td>0 &gt; 3 (6)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>9</td>
<td>16</td>
<td>56</td>
<td>127</td>
<td>2.268</td>
<td>2 &gt; 0 (32)</td>
<td>3 &gt; 2 (3)</td>
</tr>
<tr>
<td>stage5</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>10</td>
<td>24</td>
<td>61</td>
<td>2.542</td>
<td>2 &gt; 0 (20)</td>
<td>0 &gt; 3 (6)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>8</td>
<td>16</td>
<td>40</td>
<td>87</td>
<td>2.175</td>
<td>2 &gt; 0 (30)</td>
<td>3 &gt; 2 (3)</td>
</tr>
</tbody>
</table>

Table 4.1: 4-Node Dynamic programing Heuristic Approach

<table>
<thead>
<tr>
<th>DP-S</th>
<th>Node</th>
<th>Trans</th>
<th>Light</th>
<th>Wave Len</th>
<th>Phy Hop</th>
<th>Hop Wei</th>
<th>Tot Hop</th>
<th>Avg Wei</th>
<th>Max Congestion</th>
<th>Min Congestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>stage1</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>21</td>
<td>21</td>
<td>1</td>
<td>3 &gt; 2 (9)</td>
<td>2 &gt; 0 (4)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>25</td>
<td>25</td>
<td>1</td>
<td>3 &gt; 2 (9)</td>
<td>0 &gt; 3 (1)</td>
</tr>
<tr>
<td>stage2</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>5</td>
<td>22</td>
<td>46</td>
<td>2.091</td>
<td>3 &gt; 2 (9)</td>
<td>2 &gt; 0 (7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>9</td>
<td>9</td>
<td>30</td>
<td>59</td>
<td>1.967</td>
<td>0 &gt; 3 (13)</td>
<td>1 &gt; 2 (1)</td>
</tr>
<tr>
<td>stage3</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>6</td>
<td>21</td>
<td>63</td>
<td>3</td>
<td>0 &gt; 3 (11)</td>
<td>1 &gt; 2 (4)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>9</td>
<td>10</td>
<td>37</td>
<td>84</td>
<td>2.27</td>
<td>0 &gt; 3 (16)</td>
<td>3 &gt; 2 (3)</td>
</tr>
<tr>
<td>stage4</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>13</td>
<td>10</td>
<td>35</td>
<td>89</td>
<td>2.543</td>
<td>2 &gt; 0 (20)</td>
<td>0 &gt; 3 (6)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>18</td>
<td>16</td>
<td>51</td>
<td>115</td>
<td>2.255</td>
<td>2 &gt; 0 (30)</td>
<td>3 &gt; 2 (3)</td>
</tr>
<tr>
<td>stage5</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>12</td>
<td>10</td>
<td>24</td>
<td>61</td>
<td>2.542</td>
<td>2 &gt; 0 (20)</td>
<td>0 &gt; 3 (6)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>17</td>
<td>16</td>
<td>40</td>
<td>87</td>
<td>2.175</td>
<td>2 &gt; 0 (30)</td>
<td>3 &gt; 2 (3)</td>
</tr>
</tbody>
</table>

Table 4.2: 4-Node Dynamic programing Source based Approach
Table: 4.3: 4-Node Dynamic programing Destination Based Approach

By taking the above results comparison of each objective function in all stages with the proposed heuristics of Dynamic programming has been represented in the form of graphs. Each graph in its below is explained about the utilization of single objective function from stage-1 to Stage-5 with comparison of the proposed heuristics of Dynamic programming approach.

![Light paths on 4-Nodes, 2 Transceivers](image)

Fig: 4.3: Utilisation of Light paths at each stage on 4-Node Network after Implementation of DP-H, DP-S and DP-D
Fig: 4.4: Utilisation of Wavelengths at each stage on 4-Node Network after Implementation of DP-H, DP-S and DP-D

Fig: 4.5: Usage of physical hops at each stage on 4-Node Network after Implementation of DP-H, DP-S and DP-D
Fig: 4.6: Utilisation of Total hop weight at each stage on 4-Node Network after Implementation of DP-H, DP-S and DP-D

Fig: 4.7: Formation of maximum congestion at each stage on 4-Node Network after Implementation of DP-H, DP-S and DP-D
4.3 Implementation of Dynamic programming on 14-Node (NSFNET) in all pair shortest path

Fig: 4.8: 14-Node (NSFNET)

Fig 4.8: 14-Node (NSFNET) network is used to find the shortest path from each source to different destinations using Dynamic programing. Link connectivity networks in direct network costs are initialized randomly and represented as traffic matrix.

\[
\begin{pmatrix}
0 & 93 & 96 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
39 & 0 & \infty & 14 & 77 & \infty & \infty & \infty & \infty & \infty & 44 & \infty & \infty & \infty \\
78 & \infty & 0 & \infty & 39 & \infty & \infty & \infty & \infty & \infty & 88 & 98 & \infty & \infty \\
\infty & 5 & \infty & 0 & \infty & 39 & \infty & 25 & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & 26 & \infty & 0 & \infty & 28 & \infty & 23 & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & 7 & \infty & 0 & \infty & 97 & 40 & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & 61 & \infty & \infty & 0 & \infty & 36 & \infty & \infty & \infty & \infty & 42 \\
\infty & \infty & \infty & 61 & 49 & \infty & 0 & \infty & 17 & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & 12 & \infty & 57 & \infty & 0 & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & 42 & \infty & 97 & 47 & \infty & 0 & \infty & \infty & \infty & \infty & \infty \\
\infty & 29 & \infty & \infty & \infty & \infty & 0 & \infty & 41 & 28 & \infty & \infty & \infty & \infty \\
\infty & \infty & 54 & \infty & \infty & \infty & \infty & \infty & 67 & 0 & \infty & 87 & \infty & \infty \\
\infty & \infty & 26 & \infty & \infty & \infty & \infty & \infty & \infty & 81 & \infty & 0 & 3 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 4 & 58 & 0 & \infty \\
\end{pmatrix}

14-Node (NSFNET) Traffic matrix
In the above 14-Node (NSFNET) traffic matrix is represented, the network is treated as the directed network and is initialized its cost randomly and represented as a traffic matrix. It will check the neighbors on supported physical network, referred to as stage-1. That means there is no intermediate node at stage-1.

Stage-2

In the above 14-Node (NSFNET) traffic matrix one intermediate node is added in all directions forms the shortest path referred as stage-2. The marked values are current best (least-cost) values.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>93</th>
<th>96</th>
<th>107</th>
<th>170</th>
<th>135</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>137</th>
<th>184</th>
<th>194</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>39</td>
<td>0</td>
<td>∞</td>
<td>14</td>
<td>77</td>
<td>∞</td>
<td>53</td>
<td>105</td>
<td>39</td>
<td>100</td>
<td>44</td>
<td>85</td>
<td>72</td>
</tr>
<tr>
<td>78</td>
<td>171</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
<td>39</td>
<td>∞</td>
<td>136</td>
<td>79</td>
<td>∞</td>
<td>155</td>
<td>88</td>
<td>98</td>
<td>101</td>
</tr>
<tr>
<td>44</td>
<td>5</td>
<td>∞</td>
<td>0</td>
<td>82</td>
<td>82</td>
<td>39</td>
<td>∞</td>
<td>25</td>
<td>75</td>
<td>49</td>
<td>∞</td>
<td>∞</td>
<td>81</td>
</tr>
<tr>
<td>65</td>
<td>26</td>
<td>∞</td>
<td>40</td>
<td>0</td>
<td>77</td>
<td>∞</td>
<td>28</td>
<td>23</td>
<td>∞</td>
<td>70</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>85</td>
<td>∞</td>
<td>7</td>
<td>52</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>97</td>
<td>40</td>
<td>114</td>
<td>∞</td>
<td>95</td>
<td>105</td>
<td>∞</td>
</tr>
<tr>
<td></td>
<td>∞</td>
<td>66</td>
<td>61</td>
<td>78</td>
<td>∞</td>
<td>0</td>
<td>83</td>
<td>86</td>
<td>36</td>
<td>∞</td>
<td>46</td>
<td>100</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>∞</td>
<td>87</td>
<td>56</td>
<td>59</td>
<td>49</td>
<td>114</td>
<td>0</td>
<td>89</td>
<td>17</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td></td>
<td>∞</td>
<td>17</td>
<td>64</td>
<td>12</td>
<td>∞</td>
<td>57</td>
<td>51</td>
<td>164</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td></td>
<td>∞</td>
<td>68</td>
<td>∞</td>
<td>136</td>
<td>42</td>
<td>96</td>
<td>97</td>
<td>47</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td>68</td>
<td>29</td>
<td>54</td>
<td>43</td>
<td>106</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>41</td>
<td>28</td>
<td>31</td>
</tr>
<tr>
<td>132</td>
<td>96</td>
<td>54</td>
<td>∞</td>
<td>∞</td>
<td>93</td>
<td>140</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>67</td>
<td>0</td>
<td>145</td>
<td>87</td>
</tr>
<tr>
<td>104</td>
<td>∞</td>
<td>26</td>
<td>∞</td>
<td>∞</td>
<td>65</td>
<td>56</td>
<td>∞</td>
<td>∞</td>
<td>81</td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>∞</td>
</tr>
<tr>
<td></td>
<td>∞</td>
<td>58</td>
<td>114</td>
<td>∞</td>
<td>53</td>
<td>∞</td>
<td>89</td>
<td>71</td>
<td>4</td>
<td>58</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>
In the above 14-Node (NSFNET) traffic matrix, two intermediate nodes are added in all direction forms the shortest path referred as stage-3. The marked values are current best (least-cost) values.

### Stage-3

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>93</th>
<th>96</th>
<th>107</th>
<th>170</th>
<th>135</th>
<th>146</th>
<th>198</th>
<th>132</th>
<th>193</th>
<th>137</th>
<th>184</th>
<th>165</th>
<th>∞</th>
</tr>
</thead>
<tbody>
<tr>
<td>39</td>
<td>0</td>
<td>98</td>
<td>14</td>
<td>77</td>
<td>96</td>
<td>53</td>
<td>105</td>
<td>39</td>
<td>89</td>
<td>44</td>
<td>85</td>
<td>72</td>
<td>75</td>
<td>∞</td>
</tr>
<tr>
<td>78</td>
<td>171</td>
<td>0</td>
<td>91</td>
<td>173</td>
<td>39</td>
<td>∞</td>
<td>136</td>
<td>79</td>
<td>153</td>
<td>155</td>
<td>88</td>
<td>98</td>
<td>101</td>
<td>∞</td>
</tr>
<tr>
<td>44</td>
<td>5</td>
<td>89</td>
<td>0</td>
<td>82</td>
<td>82</td>
<td>39</td>
<td>110</td>
<td>25</td>
<td>75</td>
<td>49</td>
<td>90</td>
<td>77</td>
<td>81</td>
<td>∞</td>
</tr>
<tr>
<td>65</td>
<td>26</td>
<td>84</td>
<td>40</td>
<td>77</td>
<td>79</td>
<td>28</td>
<td>65</td>
<td>23</td>
<td>70</td>
<td>115</td>
<td>98</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>85</td>
<td>57</td>
<td>7</td>
<td>52</td>
<td>∞</td>
<td>0</td>
<td>91</td>
<td>97</td>
<td>40</td>
<td>114</td>
<td>∞</td>
<td>95</td>
<td>105</td>
<td>108</td>
<td>∞</td>
</tr>
<tr>
<td>105</td>
<td>66</td>
<td>100</td>
<td>61</td>
<td>78</td>
<td>132</td>
<td>0</td>
<td>83</td>
<td>86</td>
<td>36</td>
<td>100</td>
<td>46</td>
<td>100</td>
<td>42</td>
<td>∞</td>
</tr>
<tr>
<td>∞</td>
<td>85</td>
<td>56</td>
<td>∞</td>
<td>59</td>
<td>49</td>
<td>114</td>
<td>0</td>
<td>89</td>
<td>17</td>
<td>∞</td>
<td>144</td>
<td>154</td>
<td>156</td>
<td>∞</td>
</tr>
<tr>
<td>56</td>
<td>17</td>
<td>64</td>
<td>12</td>
<td>94</td>
<td>57</td>
<td>51</td>
<td>164</td>
<td>0</td>
<td>87</td>
<td>61</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>107</td>
<td>68</td>
<td>103</td>
<td>82</td>
<td>42</td>
<td>96</td>
<td>97</td>
<td>47</td>
<td>∞</td>
<td>0</td>
<td>112</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>139</td>
</tr>
<tr>
<td>68</td>
<td>29</td>
<td>54</td>
<td>43</td>
<td>106</td>
<td>93</td>
<td>82</td>
<td>134</td>
<td>68</td>
<td>∞</td>
<td>0</td>
<td>35</td>
<td>28</td>
<td>31</td>
<td>∞</td>
</tr>
<tr>
<td>132</td>
<td>96</td>
<td>54</td>
<td>110</td>
<td>173</td>
<td>93</td>
<td>140</td>
<td>190</td>
<td>133</td>
<td>176</td>
<td>67</td>
<td>0</td>
<td>95</td>
<td>87</td>
<td>∞</td>
</tr>
<tr>
<td>104</td>
<td>∞</td>
<td>26</td>
<td>∞</td>
<td>∞</td>
<td>65</td>
<td>56</td>
<td>∞</td>
<td>105</td>
<td>92</td>
<td>81</td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>∞</td>
</tr>
<tr>
<td>∞</td>
<td>100</td>
<td>58</td>
<td>114</td>
<td>131</td>
<td>97</td>
<td>53</td>
<td>136</td>
<td>∞</td>
<td>89</td>
<td>71</td>
<td>4</td>
<td>58</td>
<td>0</td>
<td>∞</td>
</tr>
</tbody>
</table>

### Stage-4

In the above 14-Node (NSFNET) traffic matrix, three intermediate nodes are added in all direction forms the shortest path referred as stage-4. The marked values are current best (least-cost) values.
In the above 14-Node (NSFNET) traffic matrix, four intermediate nodes are added in all direction forms the shortest path referred to as stage-5. These are the values for all pair shortest path (least-cost).

Fig: 4.8 is a physical network on which an initial traffic matrix is taken and treated as stage-1. Further by implementing Dynamic programming approach stage-2 to stage-5 traffic matrices are developed. The formation of virtual topologies on 14-Node (NSFNET) network are developed with the proposed heuristics namely Dynamic Programming heuristic (DP-H), Dynamic programming Source based heuristic (DP-S) and Dynamic programming Destination based heuristic (DP-D) of Dynamic programming approach.

The following tables shows the several objective functions like the utilization of Light paths, Wave Lengths, total number of Physical hops, Hop Weight, Total Hop weight, Average hop Weight, Maximum Congestion and Minimum Congestion at each stage after Implementation of the proposed heuristics known as DP-H, DP-S and DP-D of Dynamic Programming approach on 14-Node (NSFNET) traffic matrix used up to 4 transceivers
| DP-H | Node Tr | Light Wave Phy Hop Hop Wei Tot Hop Avg Wei Max Congestion Min Congestion |
|------|---------|----------------|----------|----------|----------|----------|----------|----------|----------|
| stage1 | 14 | 1 | 13 | 0 | 13 | 860 | 860 | 1 | 2 -> 12 (98) | 7 -> 9(17) |
| | 14 | 2 | 27 | 1 | 27 | 1567 | 1567 | 1 | 2 -> 12 (98) | 8 -> 3(12) |
| | 14 | 3 | 40 | 3 | 40 | 2025 | 2025 | 1 | 8 -> 5(114) | 12 - 13(3) |
| | 14 | 4 | 53 | 5 | 53 | 2601 | 2601 | 1 | 0 -> 2 (192) | 13 -> 11(4) |
| stage2 | 14 | 1 | 13 | 2 | 33 | 1572 | 4178 | 2.658 | 1 -> 0(314) | 4 -> 1(70) |
| | 14 | 2 | 27 | 6 | 66 | 3033 | 8102 | 2.671 | 3 -> 1 (506) | 11 -> 2(58) |
| | 14 | 3 | 40 | 12 | 94 | 4272 | 10754 | 2.517 | 10 - 12(645) | 1 -> 0(68) |
| | 14 | 4 | 53 | 18 | 120 | 5284 | 12742 | 2.411 | 1 -> 10 (773) | 12 - 10(81) |
| stage3 | 14 | 1 | 14 | 2 | 40 | 1627 | 4998 | 3.072 | 1 -> 4 (371) | 1 -> 0(56) |
| | 14 | 2 | 27 | 7 | 79 | 3258 | 9910 | 3.042 | 6 -> 9 (621) | 2 -> 11(95) |
| | 14 | 3 | 41 | 13 | 117 | 4813 | 14741 | 3.063 | 2 -> 5 (797) | 12 - 10(81) |
| | 14 | 4 | 55 | 22 | 152 | 6184 | 18278 | 2.956 | 1 -> 10 (992) | 12 - 10(81) |
| stage4 | 14 | 1 | 14 | 2 | 42 | 1648 | 5359 | 3.252 | 2 -> 5 (427) | 1 -> 0(56) |
| | 14 | 2 | 28 | 8 | 87 | 3282 | 11063 | 3.371 | 2 -> 5 (862) | 1 -> 0(68) |
| | 14 | 3 | 42 | 16 | 122 | 4763 | 15097 | 3.17 | 2 -> 5 (1108) | 2 -> 0(136) |
| | 14 | 4 | 56 | 25 | 159 | 6251 | 19155 | 3.064 | 2 -> 5 (1244) | 7 -> 4(129) |
| stage5 | 14 | 1 | 14 | 2 | 42 | 1648 | 5359 | 3.252 | 2 -> 5 (427) | 1 -> 0(56) |
| | 14 | 2 | 28 | 8 | 87 | 3282 | 11063 | 3.371 | 2 -> 5 (862) | 1 -> 0(68) |
| | 14 | 3 | 42 | 16 | 122 | 4763 | 15097 | 3.17 | 2 -> 5 (1108) | 2 -> 0(136) |
| | 14 | 4 | 56 | 25 | 159 | 6251 | 19155 | 3.064 | 2 -> 5 (1244) | 7 -> 4(129) |

Table 4.4: 14-Node (NSFNET) Dynamic programming heuristic based Approach
Table 4.5: 14-Node (NSFNET) Dynamic programming Source based Approach

<table>
<thead>
<tr>
<th>Stage</th>
<th>Node</th>
<th>Trans</th>
<th>Light</th>
<th>Wave Len</th>
<th>Phy Hop</th>
<th>Hop Wei</th>
<th>Tot Hop</th>
<th>Avg Wei</th>
<th>Max Congestion</th>
<th>Min Congestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>6</td>
<td>13</td>
<td>824</td>
<td>824</td>
<td>1</td>
<td>2 -&gt; 12 (98)</td>
<td>13 -&gt;11(4)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>25</td>
<td>7</td>
<td>25</td>
<td>1468</td>
<td>1468</td>
<td>1</td>
<td>2 -&gt; 12 (98)</td>
<td>8 -&gt;3(12)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>38</td>
<td>8</td>
<td>38</td>
<td>1924</td>
<td>1924</td>
<td>1</td>
<td>2 -&gt; 12 (98)</td>
<td>12 -&gt;13(3)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>42</td>
<td>9</td>
<td>42</td>
<td>2008</td>
<td>2008</td>
<td>1</td>
<td>2 -&gt; 12 (98)</td>
<td>12 -&gt;13(3)</td>
</tr>
<tr>
<td>Stage 2</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>20</td>
<td>27</td>
<td>1407</td>
<td>3212</td>
<td>2.28</td>
<td>13 -&gt; 6 (254)</td>
<td>7 -&gt;5(56)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>23</td>
<td>62</td>
<td>2850</td>
<td>7380</td>
<td>2.58</td>
<td>1 -&gt; 10 (477)</td>
<td>1 -&gt;0(68)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>28</td>
<td>91</td>
<td>4039</td>
<td>9938</td>
<td>2.46</td>
<td>10 - 12 (645)</td>
<td>1 -&gt;0(68)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>54</td>
<td>33</td>
<td>116</td>
<td>5105</td>
<td>11878</td>
<td>2.33</td>
<td>1 -&gt; 10 (773)</td>
<td>10 -&gt;11(41)</td>
</tr>
<tr>
<td>Stage 3</td>
<td>14</td>
<td>1</td>
<td>14</td>
<td>25</td>
<td>40</td>
<td>1623</td>
<td>4914</td>
<td>3.03</td>
<td>1 -&gt; 4 (371)</td>
<td>1 -&gt;0(65)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>30</td>
<td>76</td>
<td>3150</td>
<td>9528</td>
<td>3.03</td>
<td>6 -&gt; 9 (621)</td>
<td>10 -&gt;11(84)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>36</td>
<td>115</td>
<td>4599</td>
<td>14052</td>
<td>3.06</td>
<td>6 -&gt; 9 (846)</td>
<td>13 -&gt;11(100)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>51</td>
<td>44</td>
<td>145</td>
<td>5895</td>
<td>17478</td>
<td>2.97</td>
<td>1 -&gt; 10 (964)</td>
<td>12 -&gt;13(188)</td>
</tr>
<tr>
<td>Stage 4</td>
<td>14</td>
<td>1</td>
<td>14</td>
<td>28</td>
<td>42</td>
<td>1648</td>
<td>5308</td>
<td>3.22</td>
<td>2 -&gt; 5 (427)</td>
<td>1 -&gt;0(44)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>28</td>
<td>35</td>
<td>86</td>
<td>3224</td>
<td>10834</td>
<td>3.36</td>
<td>2 -&gt; 5 (955)</td>
<td>1 -&gt;0(44)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>43</td>
<td>119</td>
<td>4708</td>
<td>14938</td>
<td>3.17</td>
<td>2 -&gt; 5 (1108)</td>
<td>1 -&gt;0(105)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>53</td>
<td>52</td>
<td>149</td>
<td>6080</td>
<td>18079</td>
<td>2.97</td>
<td>2 -&gt; 5 (1244)</td>
<td>13 -&gt;12(100)</td>
</tr>
<tr>
<td>Stage 5</td>
<td>14</td>
<td>1</td>
<td>14</td>
<td>28</td>
<td>39</td>
<td>1597</td>
<td>5059</td>
<td>3.17</td>
<td>2 -&gt; 5 (427)</td>
<td>1 -&gt;10(44)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>35</td>
<td>85</td>
<td>3227</td>
<td>10881</td>
<td>3.37</td>
<td>2 -&gt; 5 (955)</td>
<td>8 -&gt;5(64)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>43</td>
<td>117</td>
<td>4660</td>
<td>14659</td>
<td>3.14</td>
<td>2 -&gt; 5 (1108)</td>
<td>3 -&gt;8(68)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>53</td>
<td>52</td>
<td>151</td>
<td>6094</td>
<td>18314</td>
<td>3.00</td>
<td>2 -&gt; 5 (1244)</td>
<td>13 -&gt;12(100)</td>
</tr>
</tbody>
</table>
Table 4.6: 14-node (NSFNET) Dynamic programming Destination based Approach

By taking the above results comparison of each objective function in all stages with the proposed heuristics of Dynamic programming has been represented in the form of graphs. Each graph in its below is explained about the utilization of single objective function from stage-1 to Stage-5 with comparison of the proposed heuristics of Dynamic programming approach.
Chapter 4: Implementation of Dynamic programming formation of Virtual Topology

Fig: 4.9: Utilisation of Light paths at each stage on 14-Node (NSFNET) Network after Implementation of DP-H, DP-S and DP-D

Fig: 4.10: Utilisation of Wave lengths at each stage on 14-Node (NSFNET) Network after Implementation of DP-H, DP-S and DP-D
Chapter 4: Implementation of Dynamic programming formation of Virtual Topology

Fig 4.11: Usage of physical hops at each stage on 14-Node (NSFNET) Network after Implementation of DP-H, DP-S and DP-D

Fig 4.12: Utilisation of total hop weight at each stage on 14-Node (NSFNET) Network after Implementation of DP-H, DP-S and DP-D
Chapter 4: Implementation of Dynamic programming formation of Virtual Topology

4.4 Dynamic Values

In Dynamic Value approach, stage wise traffic matrices are generated by keeping the maximum and minimum values of Dynamic programming approach at every stage. The least and most extreme values are taken for rearrangement of the values in random method. Dynamic value heuristic Approach (DVH), Dynamic value Source based Approach (DVS) and Dynamic value Destination based Approach (DVD) of Dynamic value approach are implemented on all stages. These heuristics are implemented on 4-Node and 14-Node (NSFNET) Network.

The DVH, DVS and DVD heuristics are used to produce objective functions like the utilisation of light paths, wavelengths, Physical hops, hop weight, total hop weight, average hop weight maximum congestion and minimum congestion. The results are represented stage wise on 4-Node and 14-Node (NSFNET) Network.

Fig 4.13: Formation of maximum congestion at each stage on 14-Node (NSFNET) Network after Implementation of DP-H, DP-S and DP-D
4.5 Implementation of Dynamic value on 4-Node in all pair shortest path

\[
\begin{bmatrix}
0 & 8 & \infty & 1 \\
\infty & 0 & 6 & \infty \\
4 & \infty & 0 & \infty \\
\infty & 2 & 9 & 0
\end{bmatrix}
\]

Stage-1

\[
\begin{bmatrix}
0 & 8 & \infty & 1 \\
\infty & 0 & 6 & \infty \\
4 & 13 & 0 & \infty \\
\infty & 2 & 9 & 0
\end{bmatrix}
\]

Stage-2

\[
\begin{bmatrix}
0 & 3 & 9 & 1 \\
5 & 0 & 6 & 7 \\
4 & 12 & 0 & 8 \\
11 & 2 & 3 & 0
\end{bmatrix}
\]

Stage-3

\[
\begin{bmatrix}
0 & 3 & 9 & 1 \\
5 & 0 & 6 & 7 \\
4 & 8 & 0 & 5 \\
11 & 2 & 3 & 0
\end{bmatrix}
\]

Stage-4

\[
\begin{bmatrix}
0 & 3 & 4 & 1 \\
5 & 0 & 5 & 6 \\
4 & 7 & 0 & 5 \\
7 & 2 & 3 & 0
\end{bmatrix}
\]

Stage-5
Fig 4.1: Physical network with initial traffic matrix treated as stage-1 on which Dynamic Value heuristic (DVH) of Dynamic Value approach is applied. The formation of stage wise 4-Node virtual topology is represented in Fig 4.14. Formation of virtual topologies may took place from stage-1 to stage-5 traffic matrices.

On the same physical network the other heuristics namely Dynamic Value Source based heuristic (DVS) and Dynamic Value Destination based heuristic (DVD) of Dynamic programming approach are implemented for formation of Virtual Topology.
The following tables show the several objective functions like the utilization of Light paths, Wave Lengths, total number of Physical hops, Hop Weight, Total Hop weight, Average hop Weight, Maximum Congestion and Minimum Congestion at each stage after Implementation of the proposed heuristics known as DVH, DVS and DVD of Dynamic Programming approach on 4-Node traffic matrix used up to 2 transceivers.

<table>
<thead>
<tr>
<th>DVH</th>
<th>Node</th>
<th>Trans</th>
<th>Light</th>
<th>Wave Len</th>
<th>Phy Hop</th>
<th>Hop Wei</th>
<th>Tot Hop</th>
<th>Avg Wei</th>
<th>Max Congestion</th>
<th>Min Congestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>stage1</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>21</td>
<td>21</td>
<td>1</td>
<td>3 -&gt; 2 (9)</td>
<td>2 -&gt;0(4)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>6</td>
<td>39</td>
<td>39</td>
<td>1</td>
<td>0 -&gt; 1 (16)</td>
<td>1 -&gt;2(6)</td>
</tr>
<tr>
<td>stage2</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>23</td>
<td>49</td>
<td>2.13</td>
<td>3 -&gt; 2 (9)</td>
<td>2 -&gt;0(7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>3</td>
<td>8</td>
<td>41</td>
<td>67</td>
<td>1.634</td>
<td>2 -&gt; 1 (11)</td>
<td>1 -&gt;2(6)</td>
</tr>
<tr>
<td>stage3</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>9</td>
<td>39</td>
<td>85</td>
<td>2.179</td>
<td>2 -&gt; 0 (25)</td>
<td>0 -&gt;1(7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>7</td>
<td>15</td>
<td>67</td>
<td>132</td>
<td>1.97</td>
<td>2 -&gt; 0 (37)</td>
<td>0 -&gt;1(10)</td>
</tr>
<tr>
<td>stage4</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>9</td>
<td>35</td>
<td>77</td>
<td>2.2</td>
<td>2 -&gt; 0 (25)</td>
<td>0 -&gt;1(7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>7</td>
<td>15</td>
<td>60</td>
<td>118</td>
<td>1.743</td>
<td>2 -&gt; 0 (37)</td>
<td>0 -&gt;1(10)</td>
</tr>
<tr>
<td>stage5</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>9</td>
<td>24</td>
<td>54</td>
<td>2.25</td>
<td>2 -&gt; 0 (24)</td>
<td>0 -&gt;1(7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>7</td>
<td>15</td>
<td>40</td>
<td>80</td>
<td>2</td>
<td>2 -&gt; 0 (38)</td>
<td>0 -&gt;1(10)</td>
</tr>
</tbody>
</table>

**Table 4.7: 4-Node Dynamic value Heuristic Approach**

<table>
<thead>
<tr>
<th>DVS</th>
<th>Node</th>
<th>Trans</th>
<th>Light</th>
<th>Wave Len</th>
<th>Phy Hop</th>
<th>Hop Wei</th>
<th>Tot Hop</th>
<th>Avg Wei</th>
<th>Max Congestion</th>
<th>Min Congestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>stage1</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>21</td>
<td>21</td>
<td>1</td>
<td>3 -&gt; 2 (9)</td>
<td>2 -&gt;0(4)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>30</td>
<td>30</td>
<td>1</td>
<td>3 -&gt; 2 (9)</td>
<td>0 -&gt;3(1)</td>
</tr>
<tr>
<td>stage2</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>23</td>
<td>49</td>
<td>2.13</td>
<td>3 -&gt; 2 (9)</td>
<td>2 -&gt;0(7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>3</td>
<td>8</td>
<td>35</td>
<td>61</td>
<td>1.743</td>
<td>2 -&gt; 0 (11)</td>
<td>1 -&gt;2(6)</td>
</tr>
<tr>
<td>stage3</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>10</td>
<td>9</td>
<td>39</td>
<td>85</td>
<td>2.179</td>
<td>2 -&gt; 0 (25)</td>
<td>0 -&gt;1(7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>14</td>
<td>12</td>
<td>50</td>
<td>95</td>
<td>1.9</td>
<td>2 -&gt; 0 (30)</td>
<td>0 -&gt;1(10)</td>
</tr>
<tr>
<td>stage4</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>10</td>
<td>9</td>
<td>35</td>
<td>77</td>
<td>2.2</td>
<td>2 -&gt; 0 (25)</td>
<td>0 -&gt;1(7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>14</td>
<td>12</td>
<td>43</td>
<td>81</td>
<td>1.884</td>
<td>2 -&gt; 0 (30)</td>
<td>0 -&gt;1(10)</td>
</tr>
<tr>
<td>stage5</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>10</td>
<td>9</td>
<td>24</td>
<td>54</td>
<td>2.25</td>
<td>2 -&gt; 0 (24)</td>
<td>0 -&gt;1(7)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>15</td>
<td>15</td>
<td>40</td>
<td>80</td>
<td>2</td>
<td>2 -&gt; 0 (38)</td>
<td>0 -&gt;1(10)</td>
</tr>
</tbody>
</table>

**Table 4.8: 4-Node Dynamic value Source based Approach**
By taking the above results, comparison of each objective function in all stages with the proposed heuristics of Dynamic programming has been represented in the form of graphs. Each graph in its below is explained about the utilization of single objective function from stage-1 to Stage-5 with comparison of the proposed heuristics of Dynamic Value approach.

![Utilisation of Light paths at each stage on 4-Node Network after Implementation of DVH, DVS and DVD](image)

**Table 4.9: 4-Node Dynamic value Destination based Approach**
Chapter 4: Implementation of Dynamic programming formation of Virtual Topology

Fig: 4.16 Utilisation of Wave lengths at each stage on 4-Node Network after Implementation of DVH, DVS and DVD

Fig: 4.17 Usage of physical hops at each stage on 4-Node Network after Implementation of DVH, DVS and DVD
Fig: 4.18 Utilisation of Total hop weight at each stage on 4-Node Network after Implementation of DVH, DVS and DVD

Fig: 4.19 Formation of maximum congestion at each stage on 4-Node Network after Implementation of DVH, DVS and DVD
4.6 Implementation of Dynamic value on 14-Node (NSFNET) in all pair shortest path

\[
\begin{pmatrix}
0 & 4 & 58 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
33 & 0 & \infty & 3 & 7 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
26 & \infty & 0 & \infty & 54 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & 87 & \infty & 0 & \infty & 29 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & 97 & \infty & \infty & 0 & \infty & \infty & 47 & \infty & 23 & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & 12 & \infty & \infty & 0 & \infty & 57 & 4 & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & 49 & \infty & \infty & 0 & \infty & 61 & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & 61 & 49 & \infty & 0 & \infty & 77 & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & 72 & \infty & 17 & \infty & \infty & 0 & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & 42 & \infty & 37 & 47 & \infty & 0 & \infty & \infty & \infty & \infty & \infty \\
\infty & 97 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 & 69 & 28 & \infty & \infty & \infty \\
\infty & \infty & 78 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 11 & 0 & \infty & \infty & 24 \\
\infty & \infty & 44 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 45 & \infty & 0 & 14 \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 96 & 39 & 0 \\
\end{pmatrix}
\]

In the above 14-Node (NSFNET) traffic matrix is represented, the network is treated as the directed network and is initialized its cost randomly and represented as a traffic matrix. It will check the neighbors on supported physical network, referred to as stage-1. That means there is no intermediate node at stage-1.

\[
\begin{pmatrix}
0 & 58 & 114 & 58 & 110 & 35 & \infty & \infty & \infty & \infty & 7 & 58 & 114 & \infty \\
104 & 0 & \infty & 14 & 77 & \infty & 53 & 26 & 39 & 100 & 81 & 85 & 3 & \infty \\
26 & 71 & 0 & \infty & \infty & 96 & \infty & 36 & 140 & \infty & 67 & 88 & 98 & 101 \\
104 & 87 & \infty & 0 & 82 & 54 & 39 & \infty & 25 & 75 & 6 & \infty & \infty & 81 \\
65 & 36 & \infty & 40 & 0 & 77 & \infty & 47 & \infty & 164 & 70 & \infty & \infty & \infty \\
7 & \infty & 105 & 52 & \infty & 0 & \infty & 57 & 40 & 70 & \infty & 95 & 105 & \infty \\
\infty & 17 & \infty & 61 & 12 & \infty & 0 & 83 & 53 & 136 & \infty & 46 & 100 & 42 \\
\infty & 81 & 56 & \infty & 61 & 114 & 78 & 0 & \infty & 89 & 87 & \infty & \infty & \infty \\
\infty & 17 & 66 & 12 & \infty & 57 & 51 & 72 & 0 & \infty & \infty & \infty & \infty & \infty \\
\infty & 136 & \infty & 66 & 78 & 108 & 97 & 47 & \infty & 0 & \infty & \infty & \infty & 100 \\
68 & 29 & 54 & \infty & 43 & 106 & \infty & \infty & \infty & \infty & 0 & 41 & 28 & 31 \\
101 & 98 & 88 & \infty & \infty & 93 & 140 & \infty & \infty & \infty & 67 & 0 & 145 & 87 \\
104 & \infty & 94 & \infty & \infty & 39 & 56 & \infty & \infty & \infty & 81 & 105 & 0 & 72 \\
\infty & \infty & 93 & 97 & \infty & 17 & \infty & 89 & 105 & 70 & 58 & 0 \\
\end{pmatrix}
\]

In the above 14-Node (NSFNET) traffic matrix one intermediate node is added in all
directions forms the shortest path referred as stage-2. The marked values are current best (least- cost) values

\[
\begin{array}{cccccccccccccccc}
0 & 58 & 96 & 5 & 110 & 35 & 146 & 53 & 132 & 71 & 7 & 58 & 14 & \infty \\
3 & 0 & 39 & 14 & 7 & 104 & 53 & 105 & 39 & 89 & 44 & 85 & 72 & 110 \\
13 & 71 & 0 & 91 & 54 & 39 & \infty & 36 & 110 & 55 & 67 & 88 & 98 & 101 \\
104 & 107 & 89 & 0 & 82 & 135 & 39 & 110 & 193 & 75 & 49 & 90 & 77 & 81 \\
65 & 154 & 84 & 40 & 0 & 77 & 79 & 47 & 28 & 65 & 70 & 115 & 98 & \infty \\
56 & 57 & 173 & 52 & \infty & 0 & 91 & 97 & 40 & 114 & \infty & 95 & 105 & \infty \\
42 & 66 & 100 & 61 & 78 & 32 & 0 & 83 & 86 & 36 & 100 & 46 & 100 & 42 \\
\infty & 100 & 56 & \infty & 59 & 49 & 78 & 0 & 17 & 89 & 87 & 114 & 25 & 54 \\
108 & 17 & 64 & 190 & 94 & 57 & 51 & 62 & 0 & 87 & 61 & \infty & \infty & \infty \\
81 & 68 & 103 & 66 & 78 & 108 & 97 & 47 & \infty & 0 & 112 & \infty & \infty & 139 \\
68 & 29 & 54 & 43 & 106 & 93 & 82 & 134 & 68 & \infty & 0 & 35 & 28 & 31 \\
101 & 98 & 88 & 110 & 3 & 93 & 40 & 12 & 133 & 176 & 67 & 0 & 95 & 87 \\
75 & \infty & 26 & \infty & \infty & 39 & 56 & \infty & 105 & 92 & 81 & 105 & 0 & 72 \\
\infty & 28 & 93 & 97 & 82 & 97 & 17 & 136 & \infty & 89 & 105 & 70 & 58 & 0 \\
\end{array}
\]

In the above 14-Node (NSFNET) traffic matrix, two intermediate nodes are added in all directions forms the shortest path it’s referred to as stage-3. The marked values are the current best (least- cost) values.

\[
\begin{array}{cccccccccccccccc}
0 & 58 & 96 & 5 & 29 & 35 & 7 & 98 & 26 & 71 & 7 & 58 & 14 & 168 \\
3 & 0 & 98 & 14 & 77 & 104 & 53 & 105 & 39 & 89 & 140 & 85 & 72 & 110 \\
78 & 71 & 0 & 91 & 54 & 39 & 130 & 36 & 110 & 55 & 67 & 88 & 98 & 101 \\
104 & 107 & 89 & 0 & 82 & 135 & 39 & 110 & 193 & 75 & 49 & 90 & 77 & 81 \\
65 & 154 & 84 & 40 & 0 & 77 & 79 & 47 & 28 & 65 & 70 & 115 & 98 & 101 \\
56 & 57 & 173 & 52 & 134 & 0 & 91 & 97 & 40 & 114 & 101 & 95 & 105 & 108 \\
42 & 66 & 100 & 61 & 78 & 32 & 0 & 83 & 86 & 36 & 100 & 46 & 100 & 42 \\
124 & 100 & 56 & 98 & 59 & 49 & 78 & 0 & 17 & 89 & 87 & 114 & 25 & 54 \\
108 & 17 & 64 & 190 & 94 & 57 & 51 & 62 & 0 & 87 & 61 & \infty & 89 & \infty \\
81 & 68 & 103 & 66 & 78 & 108 & 97 & 47 & 107 & 0 & 112 & 143 & 93 & 139 \\
68 & 29 & 54 & 43 & 106 & 93 & 82 & 134 & 68 & 118 & 0 & 35 & 28 & 31 \\
101 & 98 & 88 & 110 & 3 & 93 & 40 & 12 & 133 & 176 & 67 & 0 & 95 & 87 \\
75 & 135 & 26 & 117 & 136 & 39 & 56 & 139 & 105 & 92 & 81 & 105 & 0 & 72 \\
31 & 28 & 93 & 97 & 82 & 97 & 17 & 136 & 47 & 89 & 105 & 70 & 58 & 0 \\
\end{array}
\]

In the above 14-Node (NSFNET) traffic matrix, three intermediate nodes are added
in all directions forms the shortest path it’s referred to as stage-4. The marked values are the current best (least-cost) values.

In the above 14-Node (NSFNET) traffic matrix, four intermediate nodes are added in all directions forms the shortest path it’s referred to as stage-5. These values are the best values (least-cost) values.

Fig. 4.8 is a physical network with initial traffic matrix treated as stage-1 on which of Dynamic Value approach is applied. The formation of virtual topologies on 14-Node (NSFNET) are performed stage wise by the proposed heuristics namely Dynamic Value heuristic (DVH), Dynamic Value Source based heuristic (DVS) and Dynamic Value Destination based heuristic (DVD) of Dynamic programming approach.

The following tables shows the several objective functions like the utilization of Light paths, Wave Lengths, total number of Physical hops, Hop Weight, Total Hop weight, Average hop Weight, Maximum Congestion and Minimum Congestion at each stage after Implementation of the proposed heuristics known as DVH, DVS and DVD of Dynamic Programming approach on 14-Node (NSFNET) traffic matrix used up to 4 transceivers.
<table>
<thead>
<tr>
<th>DVH</th>
<th>Node</th>
<th>Tra</th>
<th>Light</th>
<th>Wav Len</th>
<th>Phy Hop</th>
<th>Hop Wei</th>
<th>Tot Hop</th>
<th>Avg Wei</th>
<th>Max Congestion</th>
<th>Min Congestion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>stage1</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>0</td>
<td>13</td>
<td>833</td>
<td>833</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>3 -&gt;8(25)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>26</td>
<td>1</td>
<td>26</td>
<td>1347</td>
<td>1347</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>8 -&gt;3(12)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>3</td>
<td>40</td>
<td>1757</td>
<td>1757</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>12 -&gt;13(3)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>53</td>
<td>5</td>
<td>53</td>
<td>2296</td>
<td>2296</td>
<td>1</td>
<td>0 -&gt; 2 (192)</td>
<td>13 -&gt;11(4)</td>
</tr>
<tr>
<td>stage2</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>1</td>
<td>31</td>
<td>1533</td>
<td>3930</td>
<td>2.564</td>
<td>13 -&gt; 6 (254)</td>
<td>13 -&gt;11(46)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>5</td>
<td>66</td>
<td>2894</td>
<td>7824</td>
<td>2.704</td>
<td>1 -&gt; 10 (477)</td>
<td>11 -&gt;2(58)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>10</td>
<td>95</td>
<td>4066</td>
<td>10423</td>
<td>2.563</td>
<td>1 -&gt; 10 (694)</td>
<td>12 -&gt;10(81)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>52</td>
<td>16</td>
<td>116</td>
<td>4979</td>
<td>11988</td>
<td>2.408</td>
<td>1 -&gt; 10 (766)</td>
<td>12 -&gt;10(81)</td>
</tr>
<tr>
<td>stage3</td>
<td>14</td>
<td>1</td>
<td>14</td>
<td>3</td>
<td>39</td>
<td>1606</td>
<td>4884</td>
<td>3.041</td>
<td>1 -&gt; 4 (371)</td>
<td>3 -&gt;6(51)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>7</td>
<td>76</td>
<td>3166</td>
<td>9470</td>
<td>2.991</td>
<td>2 -&gt; 5 (657)</td>
<td>10 -&gt;1(82)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>42</td>
<td>15</td>
<td>119</td>
<td>4701</td>
<td>14266</td>
<td>3.035</td>
<td>3 -&gt; 1 (847)</td>
<td>7 -&gt;5(89)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>55</td>
<td>25</td>
<td>150</td>
<td>5997</td>
<td>17544</td>
<td>2.925</td>
<td>2 -&gt; 5 (1083)</td>
<td>7 -&gt;4(85)</td>
</tr>
<tr>
<td>stage4</td>
<td>14</td>
<td>1</td>
<td>14</td>
<td>4</td>
<td>45</td>
<td>1655</td>
<td>5695</td>
<td>3.441</td>
<td>2 -&gt; 5 (520)</td>
<td>1 -&gt;0(56)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>11</td>
<td>88</td>
<td>3175</td>
<td>10865</td>
<td>3.422</td>
<td>1 -&gt; 10 (847)</td>
<td>6 -&gt;3(66)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>41</td>
<td>20</td>
<td>125</td>
<td>4676</td>
<td>14903</td>
<td>3.187</td>
<td>1 -&gt; 10 (1372)</td>
<td>13 -12(100)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>55</td>
<td>32</td>
<td>163</td>
<td>6027</td>
<td>18866</td>
<td>3.13</td>
<td>1 -&gt; 10 (1218)</td>
<td>7 -&gt;5(77)</td>
</tr>
<tr>
<td>stage5</td>
<td>14</td>
<td>1</td>
<td>14</td>
<td>3</td>
<td>44</td>
<td>1635</td>
<td>5584</td>
<td>3.415</td>
<td>2 -&gt; 5 (520)</td>
<td>3 -&gt;6(51)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>10</td>
<td>83</td>
<td>3158</td>
<td>10296</td>
<td>3.26</td>
<td>2 -&gt; 5 (806)</td>
<td>10 -&gt;1(82)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>42</td>
<td>20</td>
<td>127</td>
<td>4747</td>
<td>15186</td>
<td>3.199</td>
<td>1 -&gt;10(1218)</td>
<td>7 -&gt;5(77)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>55</td>
<td>33</td>
<td>162</td>
<td>6124</td>
<td>18911</td>
<td>3.088</td>
<td>1 -&gt; 10 (1459)</td>
<td>13 -12(100)</td>
</tr>
</tbody>
</table>

Table 4.10: 14-Node (NSFNET) Dynamic Value Heuristic Approach
<table>
<thead>
<tr>
<th>Stage</th>
<th>Node</th>
<th>Trans</th>
<th>Light</th>
<th>Wave Len</th>
<th>Phy Hop</th>
<th>Hop We</th>
<th>Tot Hop</th>
<th>Avg Wei</th>
<th>Max Congestion</th>
<th>Min Congestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage1</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>6</td>
<td>13</td>
<td>833</td>
<td>833</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>3 -&gt; 8(25)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>24</td>
<td>7</td>
<td>24</td>
<td>1248</td>
<td>1248</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>8 -&gt; 3(12)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>38</td>
<td>8</td>
<td>38</td>
<td>1701</td>
<td>1701</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>12 -&gt; 13(3)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>42</td>
<td>9</td>
<td>42</td>
<td>1754</td>
<td>1754</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>12 -&gt; 13(3)</td>
</tr>
<tr>
<td>Stage2</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>18</td>
<td>27</td>
<td>1407</td>
<td>3212</td>
<td>2.283</td>
<td>13 -&gt; 6 (254)</td>
<td>7 -&gt; 5(56)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>26</td>
<td>21</td>
<td>61</td>
<td>2766</td>
<td>7167</td>
<td>2.591</td>
<td>1 -&gt; 10 (477)</td>
<td>1 -&gt; 0(68)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>25</td>
<td>87</td>
<td>3784</td>
<td>9176</td>
<td>2.425</td>
<td>1 -&gt; 10 (694)</td>
<td>9 -&gt; 4(82)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>52</td>
<td>30</td>
<td>113</td>
<td>4799</td>
<td>11288</td>
<td>2.352</td>
<td>1 -&gt; 10 (766)</td>
<td>12 -&gt; 10(81)</td>
</tr>
<tr>
<td>Stage3</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>28</td>
<td>38</td>
<td>1529</td>
<td>4679</td>
<td>3.06</td>
<td>2 -&gt; 5 (456)</td>
<td>4 -&gt; 1(79)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>34</td>
<td>79</td>
<td>3127</td>
<td>9487</td>
<td>3.034</td>
<td>2 -&gt; 5 (657)</td>
<td>10 -&gt; 1(82)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>41</td>
<td>114</td>
<td>4380</td>
<td>13316</td>
<td>3.04</td>
<td>1 -&gt; 10 822)</td>
<td>7 -&gt; 4(85)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>53</td>
<td>51</td>
<td>145</td>
<td>5774</td>
<td>16722</td>
<td>2.896</td>
<td>2 -&gt; 5 1083)</td>
<td>7 -&gt; 4(85)</td>
</tr>
<tr>
<td>Stage4</td>
<td>14</td>
<td>1</td>
<td>14</td>
<td>36</td>
<td>44</td>
<td>1635</td>
<td>5550</td>
<td>3.394</td>
<td>2 -&gt; 5 (520)</td>
<td>3 -&gt; 6(39)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>28</td>
<td>43</td>
<td>85</td>
<td>3133</td>
<td>10352</td>
<td>3.304</td>
<td>1 -&gt; 10 847)</td>
<td>10 -&gt; 1(82)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>41</td>
<td>53</td>
<td>120</td>
<td>4579</td>
<td>14265</td>
<td>3.115</td>
<td>1 -&gt; 10 1134)</td>
<td>7 -&gt; 5(84)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>54</td>
<td>65</td>
<td>159</td>
<td>5944</td>
<td>18599</td>
<td>3.129</td>
<td>1 -&gt; 10 1372)</td>
<td>7 -&gt; 5(84)</td>
</tr>
<tr>
<td>Stage5</td>
<td>14</td>
<td>1</td>
<td>14</td>
<td>37</td>
<td>46</td>
<td>1654</td>
<td>5788</td>
<td>3.499</td>
<td>2 -&gt; 5 (520)</td>
<td>1 -&gt; 3(79)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>44</td>
<td>83</td>
<td>3185</td>
<td>10330</td>
<td>3.243</td>
<td>2 -&gt; 5 (806)</td>
<td>10 -&gt; 1(82)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>55</td>
<td>123</td>
<td>4638</td>
<td>14878</td>
<td>3.208</td>
<td>1 -&gt; 10 1302)</td>
<td>7 -&gt; 5(77)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>55</td>
<td>68</td>
<td>160</td>
<td>6000</td>
<td>18465</td>
<td>3.078</td>
<td>1 -&gt; 10 1459)</td>
<td>13 -&gt; 2(100)</td>
</tr>
</tbody>
</table>

Table 4.11: 14-Node (NSFNET) Dynamic Value Source based Approach
By taking the above results comparison of each objective function in all stages with the proposed heuristics of Dynamic value approach has been represented in the form of graphs. Each graph in its below is explained about the utilization of single objective function from stage-1 to Stage-5 with comparison of the proposed heuristic of Dynamic Value approach.

**Table 4.12: 14-Node (NSFNET) Dynamic Value Destination based Approach**

<table>
<thead>
<tr>
<th>DVD</th>
<th>Node</th>
<th>Tra</th>
<th>Light</th>
<th>Wave</th>
<th>Len</th>
<th>Phy Hop</th>
<th>Hop wei</th>
<th>Tot hop</th>
<th>Avg wei</th>
<th>Max congestion</th>
<th>Min congestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>stage1</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>9</td>
<td>13</td>
<td>554</td>
<td>554</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>12 -&gt; 13(3)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>25</td>
<td>10</td>
<td>25</td>
<td>1055</td>
<td>1055</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>12 -&gt; 13(3)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>38</td>
<td>11</td>
<td>38</td>
<td>1642</td>
<td>1642</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>12 -&gt; 13(3)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>42</td>
<td>12</td>
<td>42</td>
<td>1754</td>
<td>1754</td>
<td>1</td>
<td>5 -&gt; 7 (97)</td>
<td>12 -&gt; 13(3)</td>
<td></td>
</tr>
<tr>
<td>stage2</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>32</td>
<td>27</td>
<td>1195</td>
<td>2434</td>
<td>2.037</td>
<td>13 -&gt; 6 (254)</td>
<td>4 -&gt; 1(26)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>26</td>
<td>35</td>
<td>52</td>
<td>2171</td>
<td>4400</td>
<td>2.027</td>
<td>2 -&gt; 5 (415)</td>
<td>1 -&gt; 3(40)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>38</td>
<td>40</td>
<td>79</td>
<td>3122</td>
<td>6772</td>
<td>2.169</td>
<td>1 -&gt; 10 (559)</td>
<td>10 -&gt; 1(68)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>52</td>
<td>47</td>
<td>103</td>
<td>3863</td>
<td>8459</td>
<td>2.19</td>
<td>1 -&gt; 10 (769)</td>
<td>9 -&gt; 7(47)</td>
<td></td>
</tr>
<tr>
<td>stage3</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>53</td>
<td>34</td>
<td>662</td>
<td>1560</td>
<td>2.356</td>
<td>3 -&gt; 6 (248)</td>
<td>12 -&gt; 2(65)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>27</td>
<td>58</td>
<td>75</td>
<td>1700</td>
<td>4706</td>
<td>2.768</td>
<td>8 -&gt; 3 (494)</td>
<td>3 -&gt; 8(25)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>41</td>
<td>66</td>
<td>113</td>
<td>2946</td>
<td>7958</td>
<td>2.701</td>
<td>2 -&gt; 5 (696)</td>
<td>3 -&gt; 8(25)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>54</td>
<td>76</td>
<td>148</td>
<td>4052</td>
<td>10824</td>
<td>2.671</td>
<td>1 -&gt; 10 (1028)</td>
<td>3 -&gt; 8(64)</td>
<td></td>
</tr>
<tr>
<td>stage4</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>69</td>
<td>38</td>
<td>1048</td>
<td>2907</td>
<td>2.774</td>
<td>1 -&gt; 10 (453)</td>
<td>1 -&gt; 4(77)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>26</td>
<td>75</td>
<td>67</td>
<td>2040</td>
<td>5276</td>
<td>2.586</td>
<td>1 -&gt; 10 (593)</td>
<td>3 -&gt; 6(39)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>40</td>
<td>84</td>
<td>108</td>
<td>3308</td>
<td>9319</td>
<td>2.817</td>
<td>1 -&gt; 10 (1026)</td>
<td>3 -&gt; 8(39)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>53</td>
<td>96</td>
<td>147</td>
<td>4504</td>
<td>13222</td>
<td>2.936</td>
<td>1 -&gt; 10 (1248)</td>
<td>3 -&gt; 8(39)</td>
<td></td>
</tr>
<tr>
<td>stage5</td>
<td>14</td>
<td>1</td>
<td>14</td>
<td>71</td>
<td>40</td>
<td>1269</td>
<td>3778</td>
<td>2.977</td>
<td>2 -&gt; 5 (321)</td>
<td>1 -&gt; 4(77)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2</td>
<td>28</td>
<td>76</td>
<td>79</td>
<td>2484</td>
<td>7410</td>
<td>2.983</td>
<td>1 -&gt; 10 (643)</td>
<td>3 -&gt; 8(39)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>3</td>
<td>41</td>
<td>84</td>
<td>115</td>
<td>3716</td>
<td>11110</td>
<td>2.99</td>
<td>1 -&gt; 10 (949)</td>
<td>3 -&gt; 8(39)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>4</td>
<td>54</td>
<td>94</td>
<td>145</td>
<td>4679</td>
<td>13482</td>
<td>2.881</td>
<td>1 -&gt; 10 (1179)</td>
<td>3 -&gt; 8(39)</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 4: Implementation of Dynamic programming formation of Virtual Topology

Fig 4.20 Utilisation of Light paths at each stage on 14-Node (NSFNET) Network after Implementation of DVH, DVS and DVD

Fig 4.21: Utilisation of wave lengths at each stage on 14-Node (NSFNET) Network after Implementation of DVH, DVS and DVD
Fig: 4.22 usage of physical hops at each stage on 14-Node (NSFNET) Network after Implementation of DVH, DVS and DVD.

Fig: 4.23 Utilisation of total hop weight at each stage on 14-Node (NSFNET) Network after Implementation of DVH, DVS and DVD.
4.7 Results and analysis on Dynamic Programming heuristics:

Implementation of Dynamic programing to find all pairs shortest path produces the optimal virtual topology. The cost of the link connectivity nodes in a network is treated as directed network and represented its cost as a traffic matrix. Initial network is constructed and referred as stage-1 will be extended further from stage-2 to different stages. In stage-2, it takes one intermediate node by considering the least cost and updated to further. This process is done up to full-fledged least cost traffic matrix.

The Heuristics implemented on the traffic matrix generated through Dynamic Programming are as follows for formation of Virtual Topology.

(1) Dynamic Programming Heuristic Approach (DP-H)
(2) Dynamic Programming Source based Approach (DP-S)
(3) Dynamic Programming Destination Approach (DP-D)

Fig-4.1 & Fig- 4.8 represents the 4-Node and 14- Node (NSFNET) Network respectively, both are treated as directed networks with initial network traffic matrix is referred to as stage1. Different Stages has been implemented to obtain full-fledged
traffic matrix with shortest path from each source to different destinations. For complete network it took 5-Stages in both the Networks. Virtual topologies are developed by using Heuristic based, Source based and Destination based algorithms. The results are tabulated and representation in the graphs is made to show the effective utilization of the following objective functions.

a) Light paths,
b) Wave Lengths,
c) Physical hops,
d) Total Hop weight,
e) Maximum Congestion

Fig- 4.3, 4.4 4.5, 4.6, and 4.7 shows the effective utilization of Light paths, wavelengths, Physical hops, Total hop weight and maximum congestion on 4-Node Network at each stage. Fig- 4.9, 4.10, 4.11, 4.12 and 4.13 shows the effective utilization of Light paths, wavelengths, Physical hops, Total hop weight and maximum congestion on 14-Node Network at each stage

4.8 Results and analysis on Dynamic Value heuristics

Development of Dynamic value heuristics is from arbitrary chosen Dynamic traffic matrix generated randomly at each stage by keeping minimum value and maximum value obtained through Dynamic programming at each stage. The cost of the link connectivity nodes in this network is taken as random values and represented as its cost as a traffic matrix. Initial network is constructed and referred as stage-1 will be extended further from stage-2 to different stages. In stage-2, it takes one intermediate node by considering the least cost and updated to further. This process is done up to full-fledged least cost traffic matrix

The Heuristics implemented on the traffic matrix generated through Dynamic Value heuristics are as follows for formation of Virtual Topology

(1) Dynamic Value Heuristic Approach (DVH)
(2) Dynamic Value Source Based Approach (DVS)
(3) Dynamic Value Destination Based Approach (DVD).
The formation of Virtual Topology is done stage wise on 4-Node & 14-Node traffic matrices with Dynamic value approach by using Dynamic value Heuristic based (DVH), Dynamic value Source based (DVS) and Dynamic value Destination (DVD) based algorithms. The results are tabulated in Tables - 4.7, 4.8, and 4.9 for 4-Node network and Tables - 4.10, 4.11, and 4.12 for 14 node network. The formation of complete network took 5-stages to from full-fledged network. The graphical representation is made to show the effective utilization of the following objective functions.

a) Light paths,
b) Wave Lengths,
c) Physical hops,
d) Total Hop weight,
e) Maximum congestion,

Fig- 4.15, 4.16 4.17, 4.18, and 4.19 shows the effective utilization of Light paths, wavelengths, Physical hops, Total hop weight and maximum congestion on 4-Node Network at each stage. Fig- 4.20, 4.21, 4.22, 4.23 and 4.24 shows the effective utilization of Light paths, wavelengths, Physical hops, Total hop weight and maximum congestion on 14-Node Network at each stage.

4.9 SUMMARY

This Chapter deals with the simulation of designing of virtual topology on fiber optic networks. Input of a traffic matrix for virtual topology is one of the major criteria. In Dynamic Programming the concept of stage wise traffic matrix is used to implement the proposed heuristics namely DP-H, DP-S and DP-D. The input traffic matrix depends on the existing nodes of the Network. The experimental work is done for formation of Virtual Topology on 4-Node and 14 - Node NSFNET. For optimal Virtual Topology, it took 5-stages for complete network. To observe the effectiveness of virtual topology on fiber optic networks one more method is also adapted named as Dynamic Value Approach. The randomly generated stage wise traffic matrices are considered as Dynamic value traffic matrices, on which the heuristic DVH, DVS and
DVD are implemented. Total results are tabulated showing the effective utilization of the objective functions known as light paths, wavelength’s, Physical hops, Total hops weights, Average weighted hop, maximum and minimum congestion. The performance analysis of each objective function with respect to stages its efficiency is presented.