A new approach for Open Shortest Path Weight Setting Problem (OSPFWSP)

Ahmed Abo Ghazala
National Bank of Egypt
Communication Dept.
1178 Cornish El Nile, Cairo, EGYPT
abo_ghazala@yahoo.com

Ayman EL-SAYED, Mervat Mosa
Computer Science and Engineering Dept.
Faculty of electronic Engineering,
Menouf 32952, EGYPT
ayman.elsayed@free.fr

Abstract

Routing protocol is the nervous system to any network. It directs data to its destinations, OSPF is the famous inter-area domain routing protocol at all over the world, OSPF calculates routes as follows. Each link is assigned weights by operator. Each node in the autonomous system computes shortest paths and creates destination tables used to route data to next node on the path to its destination it directs data according to variable parameter named weights (cost), quality of routing depends on the setting of these parameters, OSPF routing is NP hard problem [1]. OSPF weights setting problem is to find a set of OSPF weights that optimizes network performance. Although a lot of trials have been made for open shortest path setting problem (OSPFWSP), no optimal setting is found. At this paper a new algorithm is developed to solve OSPFWSP, also comparison between the new algorithm and the legacy methods is done.

Key words:
OSPF, OSPFWSP, Local search, genetic algorithm, simulated annealing.

I. Introduction

The Internet has seen tremendous growth in the past decade and has now become the critical information infrastructure for both personal and business applications. It is expected to be always available as it is essential to our daily commercial, social, and cultural activities [2]. Open shortest path first is the most commonly used as an IGP intra-domain routing protocol due to their simplicity and scalability [3]. Open shortest path first is the link state routing protocols. This in particular means that some information describes link state.

The OSPF routing is based on the shortest paths computations. Each link is assigned a positive integer parameter (called weight). It is ranging from 1 to 65535 [1]. Such a parameter is set by the network administrator and it should be fixed. The lower the weight, the greater the chance that traffic will be routed on that link. Cisco, the leader router vendor assigns by default OSPF weights as inverse of the link capacity, and then the shortest paths will be the links of the higher bandwidths.

Because CISCO does not take into account traffic demands when calculates weights. This type of assigning weights does not guarantee that the network runs efficiently. Then OSPFWSP problem is determine weights to be assigned to links so as to optimize a cost function typically associated with network utilization measure.

We develop a new algorithm for solve OSPFWSP problem. The new algorithm manipulates the problems of other algorithms that are used in solving OSPFWSP problem.

Local search, genetic algorithm and simulated annealing are an examples of popular algorithms used for OSPFWSP, each algorithm have some disadvantage, the new algorithm take into account this disadvantage, and try to avoid these issues.

This paper is organized as in that section II is an explanation of general routing problem, section III is definitions of the popular algorithms that used in related works, section IV the detailed steps of the new algorithm, section V proposed solution of stuck in local minima section VI is the evaluation, section VII is bottleneck of the simulation, section VIII describing of Evaluation criteria, section IX, Experimental Results, section X The discussion, and finally section the conclusion.

II. A general routing optimizing problem

A general routing optimizing problem related to OSPF can be roughly stated as follows: Given traffic demands and a network topology, find a system of link weights such that the resulting shortest path routing is feasible within the available link capacities (i.e., that the resulting link loads do not exceed link capacities)[4].

The general routing problem can be formulated as follows. If the network is represent in a directed graph, G (N, A) whose nodes and arcs represent routers and the
links between them. Each arc (a) has a capacity C (a) which is the amount of traffic flow that the link can carry. We have a demand matrix D that for each pair S and T of nodes tells us how much traffic send from S to T where S as the source and the T as the destination of the demand. Many of the entries of demand matrix D may be zero, and in particular, D (S, T) = 0 if there is no traffic flow from S to T in G (N, A).

The general routing problem also can be defined like that there are no limitations to how the flow demand can be distributed between the paths from S to T".

In order to evaluate the effectiveness of the different solutions to OSPFWSP a linear multi-commodity flow routing problem (general routing problem) is solved with piecewise linear cost function, we can formulate the general routing problem as follow

Let L (a) is the sum over all demands of the amount of flow for that demand which is sent over link a. then utilization of link a is L(a)/C(a)

The objective function here is L (a) not excess C (a) for avoid congestion then the cost functions Φ is sum of all costs of links at G (N, A).

The cost function can be formulated as in [5]

\[ \Phi = \sum_{a \in A} \Phi_a (L(a)) \quad \text{FOR all } a \in A, \Phi_a (0) = 0 \]

This formula used in [6] it is derived experimental. We make some modification in this formula at the last line we update cost function value to be quall to

\[ \Phi (x) = n \times 500 \quad \text{for } 11/10 < x/c(a) < \infty \]

Where n is the number of links
This modification enables us to know if the congestion is occurred or not from the value of the cost function. Then the congestion can occur if

\[ \Phi > T (n \times 500) \]

Where T indicate the numbers of congested lines

This formula is used in [6] to illustrate graph between the cost function and the demand load it approximates an exponentially growing curve the figure depicts that it will be more expensive to direct traffic over loaded link as the cost function increases with increasing the traffic load and the cost increases dramatically as the utilization arrive up to 1. After that congestion can be occurs if utilization >1

Then optimal routing problem can be formulated as

\[
\min \Phi = \sum_{a \in A} \Phi_a \\
\text{subject to} \\
\sum_{x \in N} \sum_{y \in N} \sum_{a \in A} f^{(s,t)}(x,y) = D(s,t) \quad y, s, t \in N, \quad (1) \\
\ell(a) = \sum_{(s,t) \in N \times N} f^{(s,t)}_a(a) \quad a \in A, \quad (2) \\
\Phi_a \geq \ell(a) \quad a \in A, \quad (3) \\
\Phi_a \geq 3\ell(a) - \frac{2}{3} c(a) \quad a \in A, \quad (4) \\
\Phi_a \geq 10\ell(a) - \frac{4}{3} c(a) \quad a \in A, \quad (5) \\
\Phi_a \geq 7\ell(a) - \frac{12}{3} c(a) \quad a \in A, \quad (6) \\
\Phi_a \geq 500\ell(a) - \frac{1000}{3} c(a) \quad a \in A, \quad (7) \\
\Phi_a \geq 5000\ell(a) - \frac{10000}{3} c(a) \quad a \in A, \quad (8) \\
f^{(s,t)}_a(a) \geq 0 \quad a \in A; \quad s, t \in N. \quad (9)
\]

Where \( f^{(s,t)}_a(a) \) is the traffic demand that flow from node s to node t and pass on link a.

### III. Related used algorithms overview

A lot of trials have been done to solve OSPFWSP, local search, genetic algorithm and simulated annealing used to solve OSPFWS.

Local search can be used on problems that can be formulated as finding a solution maximizing or minimizing a criterion among a number of candidate
solutions. Local search algorithms move from solution to solution in the space of candidate solutions (the search space) until a solution deemed optimal is found or a time bound is elapsed. The main problem of local search is staking in local minima.

A genetic algorithm (GA) is a search technique used in computing to find exact or approximate solutions to optimization and search problems. GA is often gets stuck in suboptimal local minima.

Simulated Annealing is a generic probabilistic meta-algorithm for the global optimization problem, namely locating a good approximation to the global optimum of a given function in a large search space. It depends on the size of the network [7].

The problem that faces local search and genetic algorithm is local minima Local minima is a problem which search algorithm stuck in a non optimal solutions.

Suppose the small network have 14 link then we have to select between $28^{28}$ solution about $3 \times 10^{40}$ it is a enormous numbers of solutions.

The new algorithm begins to start updating weights array to obtain a good setting to minimize cost function. We believe that the speed and the results of the algorithm are determined by how to updates the weight array we perform three steps for complete the new algorithm. All steps will be introduced at the next section.

Empty bandwidth (EM)

Empty bandwidth or (residual bandwidth) is the first attempt for updating the weights it updates weight according to this forum.

$$E_a = C_a - L_a$$

Where $E_a$ is the empty bandwidth of the link, and $C_a$ the capacity of the link it is equal to the amount of traffic that can pass across the link without any congestion occurred, and $L_a$ (load) is the amount of traffic that acutely pass the link,

The empty bandwidth (unused bandwidth) where the weight of links that have width bandwidth is decreased and the weight of the link that has narrow bandwidth or congestion is increased.

This method prefers to send data along the link that have emptier bandwidth than the link that have congestion or less bandwidth.

This method gives us more improvement in case networks have variable bandwidth links than using this formula.

$$Q_a = L_a / C_a$$

Where $Q_a$ is the utilization of the link and $L_a$, $C_a$ as defined before, thus the new algorithm tends to direct data on the line that have more empty bandwidth it different than other algorithm, other algorithm prefer to direct data on the line that have less utilization but not in all case the line that have less utilization have also more empty bandwidth

In some cases we found that using utilization for updates weights give us more improvement results it depends on the variable sizes of the links we think for another method to updates weights

Utilization & empty bandwidth (UE)

This method is a mixed method for updates weights. Utilization of the link method is mixed with empty bandwidth method the two methods are done in parallel to take advantage of the both then we have four things for updates

1) Increase links with high utilization (U+)
2) Decrease links with low utilization (U-)
3) Decrease links with emptier bandwidth (E+)
4) Increase links with less bandwidth (E-)

We have 16 probability, the 16 probability is tested and we found that this probability give us the best result

1) Increase links with high utilization (U+)
2) Decrease links with low utilization (U-)
3) Increase links with less bandwidth (E+)

Weights are updated according these three ways, EM method still give more improvement than UE method in some cases.

To avoid local minima problem that appear in local search, the new algorithm accept any solution but keep only the best one. In that it use any new solution as a neighbor for updating regardless this solution is good or badness than the pervious one these method eliminate stuck in local minima.

V. Diversification

The main function of diversification is to escape from regions that have been discovered to another search regions as yet not discovered. After thirty iterations, we observe that EM method and UE method stuck in local minima and continue stuck then we have to have a technique for move to another region and escape this valley.

The new algorithm has this technique in that after the algorithm performs 30 iteration it start to generate weights randomly this action move the solution to another region from the search space.

Network simulator2 (NS2) is chosen for implementing the new algorithm. It is a famous network simulator at all over the world, this simulator has been tested before and can be trusted, and the simulator can be installed on two platform windows XP with help of CGYWIN and Linux also.

VII. Bottleneck of the simulation

The Bottleneck of any simulation is how to test your new setting in this case any Computing the OSPF routing resulting from a given setting of the weights turned out to be the computational bottleneck of the proposal, as many different solutions are evaluated during a convergence.

Flow chart figure 3, show the steps of the evaluation process the evaluations contains four procedures.

a) Generate Topology procedure
First step in script is to generate a topology for prepare test bed network. GT-ITM based on a model of Calvert, Bhattacharjee, Daor, and Zegura is used. GT-ITM is a collection of software tools for creation, manipulation, and analysis of graph models of internet topology [9].

b) Generate traffic procedure
To generate traffic between pairs of nodes the formula below is used, the traffic between node X and node Y can be computed according to this formula.

\[ \alpha O_x D_y C_{(x,y)} e^{-\delta(X,Y)/2\Delta} \]

Where, for each node "X", "Y", "O_x" and, "D_y" are picked as a two random numbers from [0,1]. Further, for each pair of nodes also a random number is picked as C (x,y) from [0,1].
large variation in the demands. The factor $e^{-\delta(X,Y)2\Delta}$ implies that we have relatively more demand between close pairs of nodes.

**C) Record traffic procedure**

Record traffic procedure is the important procedure in the TCL script. It execute two main tasks, first is compute traffic load at all the links of the network, second task is use this measurements of traffic load to evaluate the proposal by compute cost function and utilization of the network this formula has been used for compute utilization

$$Q_a = \frac{L_a}{C_a}$$

$$Q = \sum_{a=1}^{n} Q_a$$

$$U = \frac{Q}{N}$$

Where

- $L_a$ is the traffic load on link $a$
- $C_a$ is the bandwidth of link $a$
- $U$ is the utilization
- $N$ number of links

For compute the cost function the formula that stated in [5] is used. Also empty bandwidth is computed for all links

**D) Update weights procedure**

Updates weight procedure used to updates weights values of links according to the measurements of empty bandwidth and, utilization. It takes the new values and apply those values to the links after that the traffic is generated again to be distributed according to new setting of the weights.

**VIII. Evaluation criteria**

To evaluate the proposal three different weight setting method is chosen

Cost function and utilization is measured after running the new algorithm in different traffic amount. The results are compared with three weight setting methods.

**A) Cisco weight setting**

Cisco routers default link costs, which are inversely proportional to link capacity [3]. Cisco recommended that set weights values equal to

$$W_a = 10^8/C_a$$

Where $C_a$ the bandwidth of link $a$

**B) Unity weight setting**

Unity weight setting mean put all weights values equal to one. It means take

$$W_a = 1$$

For all $a$ from 1 to $A$ and $A$ is the numbers of links

**C) Random weight setting**

Random weight setting means setting the values of the weights in randomly way. All values is setting from 1 to $A$ where

$$n = 2 * A$$

**IX. Experimental Results**

Two network is tested (6 node with 7 link) and (10 node with 14 links) table 1 and table 2, shown the measurements of random, Cisco, unity, EM, UE and the new algorithm the results indicate that the new algorithm perform better than all methods. The arrangement of the methods from worse to best is (unit, random, Cisco, EM, UE, NA).

**X. Discussion**

In 6 node 7 link, regard to the utilization of the link we found that EM method is better than all legacy method (Cisco, unit, random), UE method is better than EM method.

NA algorithm is better than all pervious methods at link utilization comparison and also in cost function comparison, we observe that NA have a large improvement in cost function this means that, it prevents the network from congestion and this is considering the important aim we work to achieve.

In (10 node 15 link) network the topology is large than the pervious one we found that the improvement not clear like (6 node 7 link) network but this due we have limited instance of traffic generation. This limitation is due to the limited resource of the PC which, is used in implementation. We suggest that if a lot of traffic instance is generated on the test bed network the result will be much better than is gotten.

In cost function comparison we found that NA algorithm give us the best results than EM and UE also all are better than legacy methods

Reduces cost function mean good distribution of data along all links of the network this confirm our theory about the results of the second in that the results will be improved if more instance of traffic is generated.

**XI. Conclusion**

At this paper we introduce a new algorithm for OSPFWSP. This algorithm work on manipulate the problem that the other algorithms stuck in it like local minima problem, also we introduce a new method for updates weights to have fast convergence it depends on mixing between empty bandwidth and link utilization.

**XII. Reference**


Table 1. 6 node 7 link traffic amount with link utilization

<table>
<thead>
<tr>
<th>Traffic amount</th>
<th>22657</th>
<th>45314</th>
<th>67971</th>
<th>90628</th>
<th>101956.5</th>
<th>113285</th>
<th>138207.7</th>
<th>137209.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0.11</td>
<td>0.22</td>
<td>0.33</td>
<td>0.44</td>
<td>0.49</td>
<td>0.55</td>
<td>0.67</td>
<td>0.76</td>
</tr>
<tr>
<td>Cisco</td>
<td>0.089</td>
<td>0.17</td>
<td>0.26</td>
<td>0.35</td>
<td>0.4</td>
<td>0.44</td>
<td>0.54</td>
<td>0.61</td>
</tr>
<tr>
<td>EM</td>
<td>0.084</td>
<td>0.18</td>
<td>0.27</td>
<td>0.33</td>
<td>0.37</td>
<td>0.42</td>
<td>0.51</td>
<td>0.58</td>
</tr>
<tr>
<td>Unit</td>
<td>0.138</td>
<td>0.276</td>
<td>0.41</td>
<td>0.55</td>
<td>0.61</td>
<td>0.67</td>
<td>0.81</td>
<td>0.9</td>
</tr>
<tr>
<td>UE</td>
<td>0.08</td>
<td>0.16</td>
<td>0.24</td>
<td>0.32</td>
<td>0.36</td>
<td>0.4</td>
<td>0.49</td>
<td>0.55</td>
</tr>
<tr>
<td>NA</td>
<td>0.068</td>
<td>0.128</td>
<td>0.192</td>
<td>0.25</td>
<td>0.3</td>
<td>0.33</td>
<td>0.4</td>
<td>0.42</td>
</tr>
</tbody>
</table>

Table 2. 6 node 7 link traffic amount with link utilization

<table>
<thead>
<tr>
<th>Traffic amount</th>
<th>22657</th>
<th>45314</th>
<th>67971</th>
<th>90628</th>
<th>101956.5</th>
<th>113285</th>
<th>138207.7</th>
<th>137209.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random cost</td>
<td>7</td>
<td>9</td>
<td>15</td>
<td>24</td>
<td>91</td>
<td>551</td>
<td>5095</td>
<td>10092</td>
</tr>
<tr>
<td>Cisco cost</td>
<td>7</td>
<td>7</td>
<td>11</td>
<td>15</td>
<td>29</td>
<td>29</td>
<td>586</td>
<td>5523</td>
</tr>
<tr>
<td>EM cost</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>22</td>
<td>29</td>
<td>29</td>
<td>105</td>
<td>5058</td>
</tr>
<tr>
<td>Unit cost</td>
<td>9</td>
<td>20</td>
<td>5017</td>
<td>5084</td>
<td>5084</td>
<td>5578</td>
<td>10508</td>
<td>15008</td>
</tr>
<tr>
<td>UE cost</td>
<td>7</td>
<td>7</td>
<td>11</td>
<td>15</td>
<td>15</td>
<td>29</td>
<td>43</td>
<td>223</td>
</tr>
<tr>
<td>NA cost</td>
<td>7</td>
<td>7</td>
<td>9</td>
<td>13</td>
<td>15</td>
<td>15</td>
<td>36</td>
<td>103</td>
</tr>
</tbody>
</table>

Figure 4. 6 nodes, 7 links traffic with utilization

Figure 5. 6 node, 7 links traffic with cost function

Figure 6. 10 nodes, 15 links traffic with utilization

Figure 7. 10 node, 15 links traffic with cost function