# Optimization of OSPF Weights using a Metaheuristic Based on Random Samplings

**Siavash Sheikhizadeh**

***Abstract*— OSPF is the best-known intra-domain routing protocol which is employed all over the internet. In this research, an optimization problem has been introduced for optimizing OSPF weights and after mentioning the formulation of the problem[1][2], a heuristic search method based on random samplings[3] has been implemented to solve it. Finally results on some artificial networks has been compared to computations of a linear programming algorithm.**

***Index Terms*— Linear Programming, OSPF weights, Random Sampling, Routing Protocol.**

## I. INTRODUCTION

At the end of 1980's rapid growth of the internet and development of autonomous systems revealed some defects of employed routing protocols. After invention of faster processors and declining in hardware expenses, IETF adopted the approach of designing a robust inter-domain routing protocol and researches led to creation of OSPF at 1990. Router vendors embraced OSPF and nowadays this protocol is the most common intra-domain protocol all over the internet.

OSPF is a link-state protocol in which link-state advertisements (LSA) are flooded into the network by every router. LSA packets contain the information of the links to the adjacent routers. Each router receives LSA's from all other routers and by using these information forms the topological data-base of the network. this date-base determines a directed and weighted graph whose nodes are routers and weighted arcs are the physical links between the routers. Each router computes shortest paths to all other routers and uses these paths to forward the incoming IP packets[4]. OSPF performs an hierarchical routing, means that routers are grouped into some regions which are connected by a backbone region. Hierarchical routing process includes an intra-domain routing inside the source region, an inter-domain routing inside the backbone and another intra-domain routing inside the destination region.

Computing shortest paths, OSPF routers utilize the Dijkstra algorithm and consequently suffer from time complexity $\Theta(n^2)$. Hierarchical routing decreases n, number of routers inside of the region, so reduces length of routing tables which has significant effects on efficiency of routing process in huge networks.

Manuscript received on April 26, 2012.

**Siavash Sheikhizadeh**, Zarand Higher Education Complex, Shahid Bahonar University of Kerman, Zarand, Iran, +98-342-4202034, (e-mail: ssheikhizade@yahoo.com).

## II. THE OPTIMIZATION PROBLEM

### A. OSPF Weights

The numeric assigned to each link of the OSPF network which is called weight, determines cost of using the link. Shortest paths are computed based on these weights, so weighting is the most influential parameter in the routing behavior of the protocol.

By default Cisco routers set OSPF weights in inverse proportion to the link capacity i.e. more bandwidth a link has, less it costs to use that link. Although this default weighting is quite reasonable because it increases chance of high-capacity links to be suited on shortest paths, but it generally does not guarantees the optimum routing behavior and could lead to uncontrolled congestion on high-capacity links.

For a given OSPF network (which has its own specific topology, traffic demands and link capacities) finding an optimum weighting which leads to a balanced traffic distribution on all of the links, can be proposed as an optimization problem.

### B. Formulation The Problem

A directed and weighted graph $G=(N,A,c)$ and a traffic demand matrix $D_{|N|\times|N|}$ have been given, function $c: A \to \mathrm{N}$ assigns a capacity to each arc of the graph. For each pair of $(s,t)\in N\times N$, $D(s,t)\geq 0$ describes the point to point traffic demand from *s*, the source node to *t*, the destination node. After routing all the traffic demands on shortest paths, $l_a$ characterize the traffic loaded on $a\in A$ and $\phi_a(l_a)$ shows the cost function of this loading. We want to find the optimum weighting vector *W*, which minimizes the total cost of routing which is defined as:

$$\phi(W)=\sum_{a\in A}\phi_a((l_a(W)) \qquad (1)$$

$$W=\langle w_1, w_2, \ldots, w_{|A|}\rangle$$

$$w_i \in [0, \max\ weight] \qquad i=0,1,\ldots,|A|$$

For this problem various cost functions can be used as $\phi$, for example the ratio of failed packets, but for possibility of solving the problem by linear programming the cost function of preceding researches[1][2] has been used, This cost function is presented in relation (3). The growth trend of this function is in direct proportion to the load of the link:

$$\min(\phi = \sum_{a\in A}\phi_a) \qquad (2)$$

$$\phi_a \geq l_a \qquad (3)$$

$$\begin{aligned}
&\phi_a \geq 3l_a - \frac{2}{3}c_a\\
&\phi_a \geq 10l_a - \frac{16}{3}c_a\\
&\phi_a \geq 70l_a - \frac{178}{3}c_a\\
&\phi_a \geq 500l_a - \frac{1468}{3}c_a\\
&\phi_a \geq 5000l_a - \frac{19468}{3}c_a
\end{aligned} \qquad (4)$$

$$\sum_{u:(u,v)\in A} f^{(s,t)}_{(u,v)} - \sum_{u:(v,u)\in A} f^{(s,t)}_{(v,u)} = \begin{cases} -D(s,t) & v = s \\ D(s,t) & v = t \\ 0 & otherwise \end{cases} \qquad (5)$$

$$l_a = \sum_{(s,t)\in N\times N} f^{(s,t)}_a \qquad f^{(s,t)}_a \geq 0 \qquad a\in A$$

$f^{(s,t)}_a$ is part of the traffic $D(s,t)$, which is passed through the arc *a*. Equation (4) indicates a strain that guarantees the equality between the incoming and outgoing flow from every node of the graph.

### C. Complexity of The Problem in Existence of OSPF Load-balancing

If there are more than one shortest paths between two nodes of the graph, OSPF will divide the load and forward each part through one of the shortest paths, this behavior is called load-balancing. In mathematical words for every pair of nodes $(s,t)\in N\times N$ and node *u*, located on a shortest path from *s* to *t* which is the source node of two different shortest branches from *u* to *s*:

$$f^{(s,t)}_{(u,v)} \cong f^{(s,t)}_{(u,v')} \qquad (6)$$

And for every node *u* which is not located on a shortest path from *s* to *t* :

$$f^{(s,t)}_{(u,v)} = 0 \qquad (7)$$

Load-balancing turns our optimization problem to a NP-hard problem, this fact has been proved for a sort of objective function including the cost function of this research[1][5].

If the effect of load-balancing on routing behavior of the OSPF were ignored, the introduced optimization problem would be soluble in polynomial time by linear programming method[6].

## III. APPLIED METHOD

In this effort we have used a stochastic optimization algorithm to solve the mentioned problem. Stochastic algorithms content themselves with finding a near-optimum solution instead of attempting to find the global optimum which could be very time-consuming. The algorithm of this research starts with some random samplings over the whole parameter space, then a wide promising neighborhood around the best found sample will be determined and used as the next search space. Search space will be contracted and relocated repeatedly until it converges to a local optimum.

Ye T. and et. al. have called this technic, recursive random search(RRS) and have proved efficiency, scalability and robustness of this algorithm which all suit the necessities of the problem[3].

*Efficiency*: with regards to changing atmosphere of a network e.g. the traffic demands and topological parameters, a fast algorithm is required which could find an acceptable solution before occurring noticeable changes in the network parameters.

*Scalability*: Graph of a network usually has hundreds of arcs and so each weighting vector W could have many dimensions. It's very important that the method has a good performance on a multi-dimensional cost function.

*Robustness*: Simulated behavior of the OSPF is at some variance with the real function of the protocol, It means that we just have an approximation of the cost function, so the selected method should have low sensitivity to these differences.

To show these qualities they used RRS for optimizing a Rastrigin function. This function can be considered as a simple sphere function $\sum_{i=1}^{n} x_i^2$ superimposed with noise $\sum_{i=1}^{n} A.cos(2\pi x_i)$. The level of noise is decided by the value of *A*. This function has many local minima and just one global minimum at *<0,0,…,0>* and is defined as follows:

$$f(X) = n.A + \sum_{i=1}^{n}(x_i^2 - A.\cos(2\pi x_i)) \qquad (8)$$
$$x_i \in [-10,10]$$

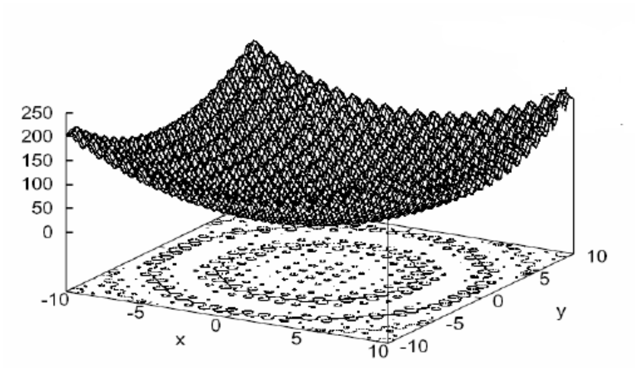

**Fig. 1: a 2-dimensional Rastrigin function**

### A. Recursive Random Search

Random sampling at the first steps has a great efficiency in finding acceptable points of the parameter space. Suppose that *f(x)* is an objective function over the parameter space *D* for a minimization problem and is bounded as:

$$y_{\min} \leq f(x) \leq y_{\max} \qquad (9)$$

The proportion of number of points in which the cost function is less than $y_r$, to entire size of the parameter space is defined as:

*Published By:*
*Blue Eyes Intelligence Engineering*
*& Sciences Publication*

$$\varphi_D(y_r) = \frac{|\{x \in D \mid f(x) \le y_r\}|}{|D|} \tag{10}$$
$$\varphi(y_{\min}) = 0 \quad , \quad \varphi(y_{\max}) = 1$$

If $\varphi(y_r) = r$ the region with size of $r.|D|$ which contains some of the best points of the parameter space and we call it r-elite area is defined as:

$$A_D(r) = \{x \in D \mid f(x) \le y_r\} \tag{11}$$
$$A_D(1) = D$$

Suppose that $n$ random samples $X_1, X_2, ..., X_n$ are taken from the parameter space and the best of them is called $X_0$, then the probability for $X_0$ to lie into r-elite region is:

$$p[X_0 \in A_D(r)] = 1 - (1-r)^n \tag{12}$$

because the probability that none of the samples belong to r-elite region is $(1-r)^n$ so:

$$r = 1 - (1-p)^{\frac{1}{n}} \tag{13}$$

Equation (13) implies that for a given $p<1$ by increasing number of samplings, $r$ will approach to zero and r-elite area will be narrowed to a small set including the global optimum and some local optimums. Contraction of r-elite area has an exponential relation with $n$.

In Fig. 2, trend of declining in $r$ by increasing $n$ has been shown for $p=0.99$. This chart shows that regardless of the cost function, only after 44 samplings we will reach to a point in 0.1-elite area with the probability of 0.99.

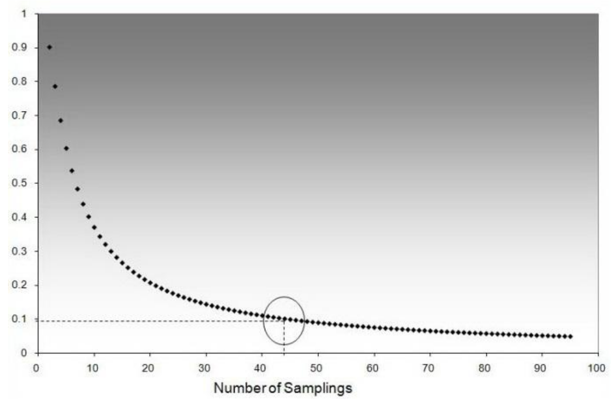

**Fig. 2, relation between n and r for p=0.99**

According to Equation (13), to reach a point which lies in the r-elite area with the probability of $p$, the number of needed samplings is:

$$n = \ln(1-p)/\ln(1-r) \tag{14}$$

The RRS algorithm at first selects two fair amounts for $r$ and $p$ and then computes $n$, the number of initial samplings from Equation (14). Acceptable amount for $r$ and $p$ are those which lead to limited number of samplings with respect to time limitation of the problem. RRS performs $n$ samplings and for the best sample $X_0$, defines the amount of its cost funtion as $f_\gamma$. This value in the next phases will be the threshold decides better samples, this step is called exploration.

Then the exploitation phase is launched. At first the algorithm selects two new amounts for $r$ and $p$ to compute a new amount for $n$, and performs at most $n$ samplings from a neighborhood of $X_0$ which dominates the r-elite area. This neighborhood is defined as follows:

$$N_{D,r}(X) = \{Z \in D \mid |z_i - x_i| < r^{1/k} \cdot (u_i - l_i)\} \tag{15}$$
$$X = < x_1, x_2, ..., x_k > \quad , \quad l_i \le x_i \le u_i$$

During an exploitation phase when a better sample is found it will be replaced as the new $X_0$, and its cost is defined as $f_\gamma$. Then the exploitation will be continued inside the neighborhood of the new $X_0$, this action is called re-align sub-phase. If after $n$ samplings no better sample has been found, neighborhood will be narrowed more by reducing amount of $r$, this action is called shrink sub-phase.

In Fig. 3 shrink and re-align sub-phases have been illustrated on a two-dimensional parameter space. A primary sample C1 has been found during the first exploration and after sampling around it inside the neighborhood R1, the better sample C2 has been found and the neighborhood has been moved to R2 around C2. After exploiting inside R2 and not finding any better sample, sampling space has been shrunk to R3 and then to R4.

shrink and re-align sub-phases will be done many times until the neighborhood reaches to a specific size, then the best sample inside the last neighborhood will be introduced as the final solution.

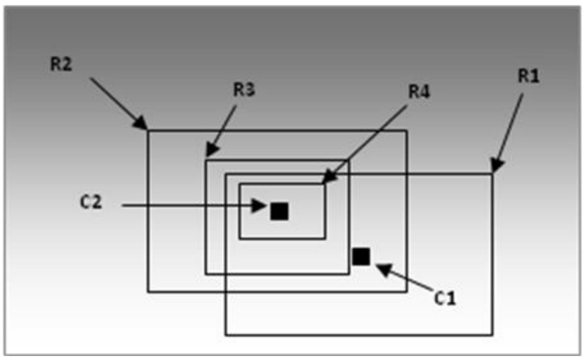

**Fig. 3, shrink and re-align sub-phases.**

## *B. Parameter Space*

For our optimization problem points of the parameter space are |A|-dimensional vectors as:

$$W = \langle w_1, w_2, ..., w_{|A|} \rangle \qquad w_i \in [0, \text{maxweight}] \tag{16}$$

Size of the parameter space is $|D| = maxweight^{|A|}$, *maxweight* is the maximum weight could be assigned to links.

Selecting a big amounts for *maxweight* expands the parameter space significantly and will decrease the efficiency of the algorithm, in this research just same as in the research done by Fortz B. et. al. [1], we've set *maxweight=20*.

In the presented algorithm for the exploration phase, $r$ and $p$ have set to 0.1 and 0.99 respectively. In exploitation phase these amounts could be varied according to the time limitations of the problem. Results show that for all the artificial networks used in this research, if number of samplings in each exploitation phase is less than 70, the run-time of the algorithm won't exceed 5 minutes.

The neighborhood in this problem is defined as:

$$N_{D,r}(W_0) = \{W \in D \mid |w_i - w_i^0| < \sqrt[|A|]{r} \cdot \max weight\} \quad (17)$$

*C. 2-3- Cost Function*

In Link-state routing protocols whenever a topological change occurs, the adjacent routers will inform these changes by flooding a new LSA into the network. If many of weights change simultaneously, many of the routers will inject LSA 's into the network and this will noticeably increase convergence time. Convergence time is the time is needed to make all the routers aware of a new link-state change. Before a network converges, routing tables are inconsistent and this causes circular routings and extremely increases the congestion on links.

If it's possible to find an optimal weighting which slightly differs from the current weighting, it will improve the convergence time, so we've added another factor to the cost function so that it takes precedence over weightings with less differences to existing weighting.

The ratio of number of changing weights to the total number of weights has been called $\phi_w$ , then a new cost function has been introduced by adding a multiple of $\phi_w$ to the normalized cost function $\phi^*$ of the research [1], it's defined as:

(18)

$$\varphi = \varphi^* + c \, . \, \varphi_w$$

$$\varphi_w(W') = \frac{1}{|A|} \sum_{a \in A} t_a \quad , \quad W' = < w'_1, w'_2, \ldots, w'_{|A|} >$$

$$t_a = \begin{cases} 1 & if \quad w'_a \neq w_a \\ 0 & else \end{cases}$$

Coefficient *c* must be selected intelligently so that $c.\phi_w$ amounts to the optimal values of $\phi^*$ and has determining effects in equal situations.

## IV. DISCUSSION AND RESULTS

To analyze the efficiency of the implemented algorithm, it run on three different artificial networks[7]. In Charts 1, 2 and 3 the amounts of normalized cost function have been presented with respect to the summation of the traffic demands.In these charts the optimal weighting of the algorithm can be compared to InvCap, Unit and LPLB weightings. InvCap is the weighting in inverse proportion to link capacity, Unit is weighting which assign 1 to all of the links and LPLB is the optimal weighting of linear programming method. Steep climbing in charts determines the saturating point of the network links.

The first examined graph was a two-hierarchical graph with 50 nodes and 148 arcs in which for more similarity to real networks, in first level link capacities were greater than capacities in the second level. In these graphs fist level is an acyclic or semi-acyclic graph and each node itself, expands to an acyclic or semi-acyclic graph in the second level. Results of running the algorithm on this graph have been presented in Chart (1), it can inferred that the algorithm has found a weighting which has close behavior to the lpbp optimal weight, this weighting is much more better than both of invcap and unit weightings.

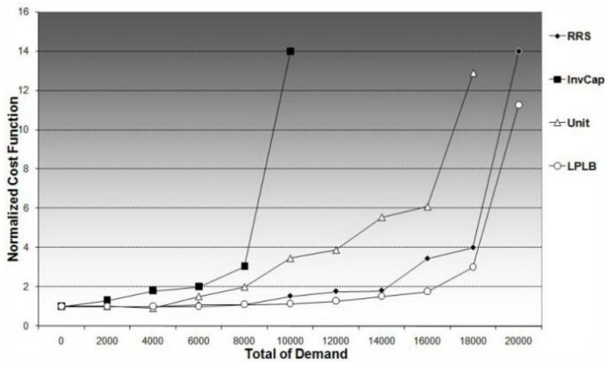

**Chart (1), Results for a 2-hierarchical graph**

The second used graph was a pure random graph with 50 nodes and 228 arcs which had no similarity to a real network unless it was connected. As shown in Chart (2) like in previous graph, corresponding chart of RRS algorithm fluctuates below two other simple weightings and is very closed to *LPLB* chart. In this case *InvCap* weighting has worse function than before because assigning random capacities to the links without considering their position could make severe congestions on some links.

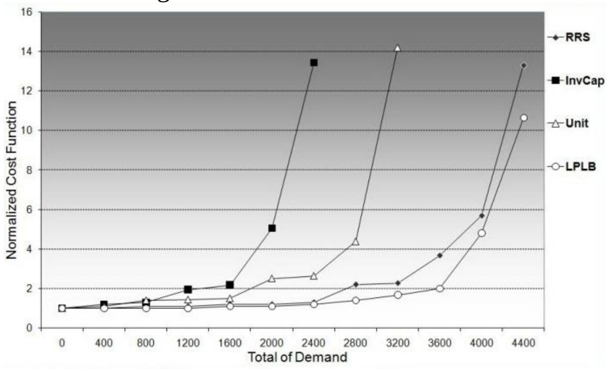

**Chart (2), Results for a pure random graph**

The third examined graph was a Waxman graph with 50 nodes and 169 arcs. Waxman graph is type of random graph

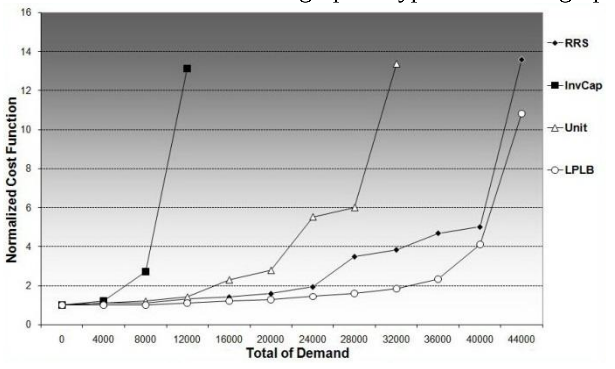

**Chart (3), Results for a Waxman graph**

in which the magnitude of link capacity between two nodes relates in inverse proportion to their Euclidian distance, so these graphs have more residence to real networks[8]. In this case the results are similar to the previous case. The RRS chart is closer to *LPLB* and *InvCap* also behaves better than before, these improvements are as the result of allocating more rational capacities to the links.

In the 2-hierarchical graph which has the most similarity to a real network, RRS algorithm makes it possible to route from 32 upto 43 percent of more traffic, this is inferred from the horizontal distance between RRS and other two simple weightings at the saturation point. This growth is between 30 and 66 percent for the pure random graph and around 65 percent for the Waxman graph.
Increasing network capacities proportional to the number of users is an unattainable or at least a costly approach. Traffic engineering could help us to save network resources and use them in the most optimal manner, the goal which presented algorithm could provide it in OSPF networks.

## REFERENCES

1. B. Fortz, M. Thorup, “Increasing internet capacity using local search”, Technical Report, AT&T Labs Research, 2000.
2. M. Ericsson, M.G.C Resende, P.M. Pardalos, " A genetic algorithm for the weight setting problem in OSPF routing ", Journal of Combinatorial Optimization, 2001.
3. T. Ye, S. KalyanaramanPoor, " A recursive random search algorithm for optimization network protocol parameters ", Technical Report, ECSE Department, Rensselaer Polytechnic Institue, 2001.
4. J. T. Moy, *OSPF Anatomy of an internet routing protocol*, Addison-Wesley, 1998.
5. B. Fortz, M. Thorup, "Internet traffic engineering by optimizing OSPF weights", IEEE INFOCOM – The Conference on Computer Communications, 2000.
6. L. G. Khachiyan, "A polynomial time algorithm for linear programming", Dokl. Akad. Nauk SSSR 244, 1979.
7. E. W. Zegura, GT-ITM Georgia tech internetwork topology models, 1996.
8. B. M. Waxman, "Routing of multipoint connections", IEEE Journal, Selected area in communications(Special issue on broadband packet communications),6(9), 1617-1622, 1998.

## AUTHORS PROFILE

**Siavash Sheikhizadeh,** M.Sc. in Computer Science, Sharif University of Technology, Mathematical Science Department, Tehran, Iran.
Faculty member of Shahid Bahonar University of Kerman, Kerman, Iran .