A Review of Optimization and Intelligence Approaches for Traffic Engineering in IP Network

Pongsarun Boonyopakorn* and Phayung Meesad**

Abstract

Traffic Engineering requires efficient tools to optimize network performance and traffic delivery. Most researches focus on routing optimization to find the shortest path length and balance load bandwidth in a network. In this paper we review the optimization approaches for Internet traffic engineering such as network routing optimization. We also review the intelligence approaches which are popular in this area such as Ants Colony Optimization (ACO), Practical Swarm Optimization (PSO), and other intelligence techniques. In addition, we summarize the applied techniques used for network optimization.

Keywords: Traffic Engineering, Ants Colony, Praticle Swarm Optimization, and Genetic Algorithm.

1. Introduction

The Internet today provides a collection of routers and links managed by Internet Service Providers (ISP). ISP consists of a routing path between any node pair (routers) that limits the throughput achievable between them. Traffic Engineering (TE) implements strategies for a good QoS achieve operational efficiencies and differentiate to their service offerings. TE is defined as that aspect of Internet network engineering dealing with the issue of performance evaluation and performance optimization of operational IP networks. The goal of performance optimization of operational IP networks is accomplished by routing traffic in a way to utilize network resources efficiently and reliably. TE has been used to imply many problems such as load balancing, constraint-based routing, multi-path routing, and fast re-routing. One of research area is single-path routing which can be effectively used for maximum utilization of network resources. The various algorithms discussed give solutions for effectively calculating the single-path and way to minimize delay and increase throughput. We surveyed the various techniques for traffic engineering. Especially, these works can be applied to IP network, then enhance network performance through traffic engineering to meet the QoS requirements.

We can classify the task of TE into two concepts which are intra-domain and inter-domain. The main task of the intra-domain is to optimize customer traffic routing between AS border routers within a single domain. While inter-domain mainly focuses on how to select AS border router (ASBR) optimally as the ingress/egress points for inter-domain traffic that travels across the local AS.

Before we go to the next session, we should introduce shortest path problems which are the main idea of TE. Shortest path problems are among the fundamental problems studied in computational geometry, also other areas including graph algorithms, network optimization and geographical information systems (GIS). Given two points s and d on the surface of a polyhedron, find the shortest path on the surface from s (node 6) to *d* (node10). Given source nodes in a weighted directed graph *G*, shown in Fig 1, with *n* nodes and *m* arcs, the shortest path problem from *s* is finding the minimum weight paths from s to all other nodes of *G*.

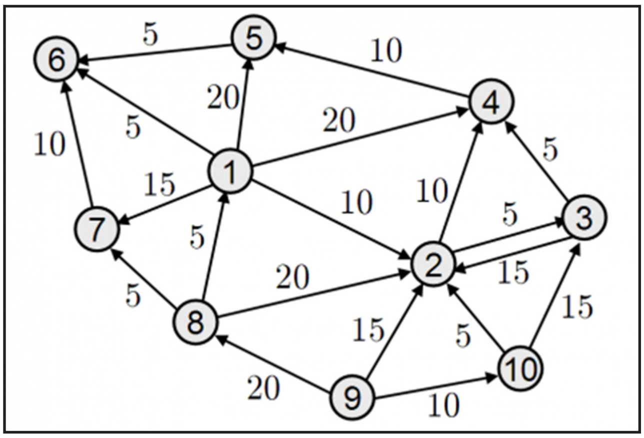

Figure 1 *Graph.*

From Fig1, the shortest path problem, finding the path with minimum distance, time or cost from a source to a destination, is one of the most fundamental problems in network theory. Next we introduce the offline/online traffic engineering and tools for solving problems.

2. Offline/Online Traffic Engineering

The difference between Online and Offline algorithms are area of traffic flows which passed through node in AS. When source (ingress node) send traffic flows to destination (egress

* *Department of Information Technology Management, Faculty of Information Technology, KMUTNB.*

** *Department of Teacher Training in Eleetrical Engineering, Faculty of Technical Education, KMUTNB.*

node), source nodes wish to find feasible shortest path to send traffic flows then the gateway nodes selected by offline algorithms. Many researchers proposed several techniques to optimize traffic flows in AS, we will discussed these techniques next. Online algorithms perform by searching the optimal path to reach destination node between egresses AS. There are many problems in networking which apply this technique such as Facility Location Problems, *k*-Center Problems, and Greedy Heuristic algorithm.

A) Facility Location Problem

Facility location problems [1], [2], [3], [4], [5] are traditionally investigated with the assumption that all the clients are to provide services. A significant shortcoming of this formulation is that a few very distant clients, called outliers, can exert a disproportionately strong influence over the final solution.

Given a set of locations $N = \{1, . . . , n\}$, and distances between them, c_{ij}, $i, j = 1, . . . ,$ n; there is a subset $F \subseteq N$ of locations at which we may open a facility, and a subset $D \subseteq N$ of locations that must be assigned to some open facility; for each location $j \in D$, there is a positive integral demand dj that must be shipped to its assigned location. For each location $i \in F$, the non-negative cost of opening a facility at i is f_i. The cost of assigning location i to an open facility at j is c_{ij} per unit of demand shipped. We assume that these costs are non-negative, symmetric, and satisfy the triangle inequality: that is, $c_{ij} = c_{ji}$ for all $i, j \in N$, and cij + cjk ≥ cik for all $i, j, k \in N$. We wish to find a feasible assignment of each location in D to an open facility so as to minimize the total cost incurred.

This problem can be stated as the following integer program, where the 0 - 1 variable y_i, $i \in F$ indicates if a facility is opened at location i, and the 0 - 1 variable x_{ij} , $i \in F\, j \in D$, indicates if location j is assigned to a facility at i:

$$minimize \sum_{i \in F} f_i y_i + \sum_{i \in F} \sum_{j \in D} d_j c_i x_{ij} \qquad (1)$$

subject to

$$\sum_{i \in F} x_{ij} = 1, \qquad (2)$$

$$x_{ij} \le y_i \quad \text{for each } i \in F, j \in D \qquad (3)$$

$$x_{ij} \in \{0, 1\} \quad \text{for each } i \in F, j \in D \qquad (4)$$

$$y_i \in \{0, 1\} \quad \text{for each } i \in F \qquad (5)$$

The constraints (2) ensure that each location $j \in D$ is assigned to some location $i \in F$, and the constraints (3) ensure that whenever a location j is assigned to location i, then a facility must have been opened at i (and paid for). For notational simplicity, refer to 0 - 1 variables xij for each $i, j \in$ N, with the understanding that if $i \notin F$ or $j \notin D$, then $x_{ij} = 0$; similarly, refer to variables y_i, for each $i \notin F$, with the understanding that $y_i = 0$ in this case.

B) *k*-Center Problem

The number of centers such that the distance from any node to the nearest center is minimized, known as the minimum *k*-Center problem, is NP-complete [1]. However, if we are willing to tolerate a factor of 2 inaccuracies, i.e. the maximum distance between any node and a center is no worse than twice the maximum in the optimal case; the problem is solvable as follows:

Algorithm *k*-Center [2]
1. Construct G_1^2, G_2^2, … , G_m^2
2. Compute M_i for each G_i^2
3. Find smallest i such that $|M_i| \le K$, say j
4. M_j is the set of K centers

Figure 2 *Algorithm for the k-Center Problem.*

Given a graph $G = (V, E)$ and all its edges arranged in non-decreasing order by link cost, c: $c(e_1) \le c(e_2) \le \ldots \le c(e_m)$; let $G_i = (V, E_i)$, where $E_i = \{e_1, e_2, \ldots, e_i\}$. A square graph of G, G^2 is the graph containing and edges (u, v) wherever there is a path between u and v in G of at most two hops, $u \ne v$ hence some edges in are pseudo edges, in that they don't exist in G. An independent set of a graph $G = (V, E)$ is a subset $V' \subseteq V$ such that, for all $u, v \in V'$, the edge (u, v) is not in E. An independent set of G^2 is thus a set of nodes in G that are at least three hops apart in G. We also define a maximal independent set M as an independent set V' such that all nodes in $V - V'$ are at most one hop away from V'. The outline of the minimum *k*-Center algorithm is shown in Fig. 2. The basic observation is that the cost of the optimal solution for the *k*-Center problem is the cost of e_i, where i is the smallest index such that G_i has a dominating set of size at most K. This is true since the set of the center nodes is a dominating set, and if G_i has a dominating set of size K, then choosing this set for the centers guarantees that the distance from a center to a client is bounded by e_i. The second observation is that a star topology in G_i, transfers into a clique (full mesh) in G_i^2. Thus, a maximal independent set of size K in G_i^2 implies that there exists a set of K stars in G, such that the cost of each edge in it is bounded by $2e_i$. The smaller the i, the larger the K. The solution to the minimum . *k*-Center problem is the G_i^2 with K stars.

C) Greedy Heuristic Problem

The basic idea of the greedy algorithm is as follows [4]. Suppose we need to choose M replicas among N potential sites. We choose one replica at a time. In the first iteration, we evaluate each of the N potential sites individually to determine its suitability for hosting a replica. We compute the cost associated with each site under the assumption that accesses from all clients converge at that site, and pick the site that yields the lowest cost. In the second iteration, we search for a second replica site which, in conjunction with the site already picked, yields the lowest cost. In general, computing the cost, we assume that clients direct their accesses to the nearest replica (i.e., one that can be reached with the lowest cost). We iterate until we have chosen M replicas.

```
Algorithm Greedy Heuristic [9]
  if (|M| ≤ ℓ)
    Choose among all sets M' with |M'| = |M|
      the set M" with minimal O (M", p)
    return set M"
  end
  Set M' to be an arbitrary set of size ℓ
  while (|M'| < |M|)
    Among all sets X of ℓ elements in M'
      and among all sets Y of ℓ + 1 elements
      in V – M' + X, choose the sets X, Y
      with minimal O (M' – X + Y, p)
    M'= M' – X + Y
  end
  return set M'
```

Figure 3 *Algorithm for the Greedy Heuristic Problem.*

Many offline/online algorithms describe how to improve the network performance in IP network. Most papers based on performance average of lengths between routes [1] – [11]. Researcher such as [6], divided the placement problem of IP network services into two sub-problems, finding the best location for a service gateway and selecting the best service gateway of the AS network service. According to [6], placement has been provided in way of dividing it into a sub-problem: to find the location gateway and select these service gateways. They propose efficient algorithms for both sub-problems and study performance. Researcher [2] is interested in the placement of cache location, they studied the cache problem with an emphasis on transparent end-route caches only located along routes from clients to servers. Researcher [3], [10] studied proxy location, mirror placement is popular in this area. Another paper such as [4], studied the performance improvements as the number of mirrors increases under different placement algorithms subject to the constraint that mirrors can be placed only at certain locations. Other papers focusing on placement via web server replicas [5], develop a placement algorithm for workload information, such as client latency and request rates, to make informed placement decisions.

3. Artificial Intelligence Approaches for Network Traffic Engineering

We found that many techniques have been presented to find the minimum optimize traffic in AS. But they only considered link cost capacity, the number of hop counts, and network delay. The advent of low cost computing devices has led to the explosive growth in the area of computer networks. There are several benefits to be achieved by this dispersal of computing across a network such as resource sharing and improved reliability. One of the major advantages of computer networks over the centralized systems is their potential for improved system reliability. The reliability of a system depends not only on the reliability of its nodes and communication links, but also on the fact how nodes are connected by communication links. A completely connected network has the highest computer network reliability, while a simple loop network has the lowest computer network reliability. In this session we will discuss parts of the intelligence approaches to design network reliability and find network optimization used with TE. There are many techniques proposed such as Genetic Algorithm, Practical Swarm Approach, and Ants Colony Optimization.

A) Genetic Algorithm

Genetic Algorithms (GA) are adaptive heuristic search algorithms which are premised on the evolutionary ideas of natural selection and gene types [14], [15]. The basic concept of GAs is designed to simulate processes in a natural system necessary for evolution, specifically those that follow the principles of survival of fittest, first laid down by Charles Darwin. As such they represent an intelligent exploitation of a random search within a defined search space to solve a problem. Basically, several random sets of parameters are applied to an algorithm and a fitness value (optimization value) is calculated for each. Based on this fitness values, the best sets are mixed (this is a combination of Selection, Crossover and Mutation) together and new sets are again applied to the algorithm until an optimal parameter(s) are obtained. This effect is usually obtained by breaking the genetic algorithm into a few small parts. We discuss the GA placement solution below:

a) Objective

Given a network of n nodes, each has a value k, the k value is cost of the shortest path where each node reaches other nodes in the network topology. This can be computed as follows

$$min \int x = \sum_{i=0}^{n} C_i \quad (6)$$

x is the node in the network topology and C is the shortest path cost of each node, we summarize the minimum path length to pick the set of node in network topology to be a service gateways where the group has optimal solutions.

b) Chromosome Structures

All nodes connected in the network will maintain the link to other nodes. Each link and node is analogous to the chromosome in the genetic algorithm. The structure of chromosome is considered to be a group as follows

```
11100111100100110110
10110111111100001110
00011100100000001110
00100101100010111010
11001010000110011110
```

Figure 4 *Binary matrix representation of solution.*

where element a_i = 1 if the node i is optimal path length in the network topology, otherwise 0. A set of solutions is randomly produced initially. By defining the neighborhood of a solution as a set of solutions having one-bit different from the original one, the optimal path length of a group of each existing solution are computed.

c) Crossover

Crossover points are randomly selected and a matching section is specified for swapping the gene of the parents. Beyond the crossover, we again use the cross point index of 20%. This will make our new varieties of generation chromosomes.

d) Sorting

We re-order the positions of the chromosomes where optimal solution is computed by fitness.

e) Mutation

The operational rates for mutations are set to 0.025 and two randomly chosen genes are then swapped.

f) Fitness

In order to calculate the summary of each path length, we need to transform the order chromosome from binary to an integer matrix. Path cost will be put into a node according to their orders in the order chromosome. We then achieve an optimal group set for service gateways in the network. Fig 5 shows the chromosome conversed to an integer.

0	1	2	...	...	...	6	...
1369	1073	1147	...	...	...	1628	...

Figure 5 *Order chromosome representations of genes.*

Each chromosome contains two strings of the same length. The first one is an ordered integer string, where each integer value represents a node ID. The second string is an ordinary integer string, where each integer value represents the number of distance of the node.

Table 1 *Parameters of GA.*

Number of generations	100
Crossover rate	Random between 0.2-0.8
Mutation rate	0.025

In [13], presents a network design problem, this paper also falls under the network topology category which is a minimum spanning tree. A strong fitness function is developed here for solving this network optimization problem which not only reduces the number of generation but also produces the best result and follows the concept of "Survival of the fittest". Fitness function is the backbone of the concept of genetic algorithms which directly affects the performance; so one of the main focuses of this paper is fitness function. In this paper, we show that genetic algorithms are an alternative solution for this NP problem where conventional deterministic methods are not able to provide the optimal solution. The researcher in [14], presented optimization algorithms for resource placement in Content Delivery Network (CDN), obtained by Greedy algorithm, as compared with Tabu search and direct-code GA. In [15], development of a genetic algorithm to solve a network routing protocol problem was proposed. The algorithm has to find the shortest path between the source and destination nodes. In the literature, the routing problem is solved using search graph techniques to find the shortest path. The developed genetic algorithm is compared with Dijkstra's algorithm to solve routing problems.

B) Practical Swarm Optimization

Particle swarm optimization is a population based stochastic optimization technique inspired by the social behavior of bird flock (and fish school etc.), as developed by Kennedy and Eberhart in 1995, [18]. The search for optimal position (solution) is performed by updating the particle velocities, hence positions, in each iteration according to the following two equations (7) and (8)

$$PV_{id} = PV_{id} + \theta_1 r_1 (B_{id} - X_{id}) + \theta_2 r_2 (B^n_{id} - X_{id});$$
$$i = 1, 2, ..., N_s \text{ and } d = 1, 2, ..., D \quad (7)$$

$$X_{id} = X_{id} + PV_{id} \quad (8)$$

where θ_1 and θ_2 are positive constants, called acceleration coefficients, N_s is the total number of particles in the swarm, D is the dimension of problem search space, i.e., number of parameters of the function being optimized, r1 and r2 are two independently generated random numbers in the range [0, 1] and "n" represents the index of the best particle in the neighborhood of a particle. The other vectors are defined as: $X_i = [X_{i1}, X_{i2}, \dots, X_{iD}]$ Position of i-th particle; $P_{Vi} = [PV_{i1}, PV_{i2}, \dots, PV_{iD}]$ Velocity of the i-th particle; $B_i = [B_{i1}, B_{i2}, \dots, B_{iD}]$ - Best position of the i-th particle ($pBest_i$), and $[B^n_{i1}, B^n_{i2}, \dots, B^n_{iD}]$ = Best position found by the neighborhood of the particle i ($nBest_i$).

Eq. (8) calculates a new velocity for each particle based on its previous velocity, the particle's position at which the best possible fitness has been achieved so far, and the neighbors' best position achieved. Eq. (7) updates each particle's position in the solution hyperspace. θ_1 and θ_2 are two learning factors, which control the influence of pBest and nBest on the search process. In all initial studies of PSO, both θ_1 and θ_2 are taken to be 2.0. However, in most cases, the velocities quickly attain very large values, especially for particles far from their global best. As a result, particles have larger position updates with particles leaving boundary of the search space. To control the increase in velocity, velocity clamping is used in Eq. (8). Thus, if the right side of Eq. (8) exceeds a specified maximum value PV^{max}_d, then the velocity on that dimension is clamped to PV^{max}_d.

In paper [18], presents the application of PSO based search algorithm for solving the single source shortest path problem (SPP) commonly encountered in graph theory. A new particle encoding/decoding scheme has been devised for representing the SPP parameters as a particle. In order to enhance the search capability of PSO, a selective local search mechanism and periodic velocity re-initialization of particles have been incorporated. Simulation results on several networks with random topologies are used to illustrate the efficiency of the proposed hybrid PSO algorithm for computation of shortest paths in networks.

C) Ants Colony Optimization

The first Ant Colony Optimization (ACO) algorithm called Ant System applied to Traveling Salesman Problem (TSP) by Dorigo, [16], [17]. It makes up the main framework of other ACO algorithms and is considered as a prototype. In TSP each of m artificial ants generates a complete tour by a probabilistic rule (1), which is the probability that ant k in city i visits city j.

$$P^k_{i,j} = \begin{cases} \dfrac{\tau^\alpha_{i,j}.\eta^\beta_{i,j}}{\sum_{j \in N^k_j} \tau^\alpha_{i,j}.\eta^\beta_{i,j}} & j \in N^k_j \end{cases} \quad (9)$$

where $\tau_{i,j}$ is pheromone, $\eta_{i,j}$ is heuristic function and is equal to $\frac{1}{d_{i,j}}$, the inverse of the difference between city i and j, N^k_j is the set of cities that haven't been visited by ant k, α and β are parameters which shows the relative importance of pheromone versus heuristic or exploitation versus exploration.

Eq. (9) shows that ants prefer paths with shorter length and higher amount of pheromone, so they independently generate tours by pre-knowledge of the problem and cooperative informative communication. Once all the ants complete their tours the pheromone trails updates, using (10) and (11)

$$\tau_{i,j} = (1-p)\tau_{i,j} + \sum_{k=1}^{m} \Delta\tau^k_{i,j} \quad (10)$$

$$\Delta\tau^k_{i,j} = \begin{cases} \dfrac{Q}{L_k} & \textit{the ant k pases link } (i, j) \end{cases} \quad (11)$$

where ρ is evaporation rate, L_k is the length of tour taken by ant k, Q is a constant, and m is the number of ants. Pheromone evaporation is a process of decreasing the intensities of pheromone trails over time. This process is used to avoid local convergence and to explore more search space. Daemon actions are optional for ACO, and they are often used to collect useful global information by depositing additional pheromone.

There are many researchers using ACO approach to solve the shortest path optimization. Researcher in [16], presented Multi-objective ACO algorithms for shortest route problem (MACO). Firstly, the pheromone on every path segment is initialized to an initial value and ants are randomly distributed among cities. Secondly, self-adaptive operator is used, namely in prophase we use higher probability to explore more search space and to collect useful global information; otherwise in anaphase we use higher probability to accelerate convergence. MACO algorithm adopts self-adaptive operator to make the search scope reduced in anaphase, thus the search time of this algorithm is reduced greatly. In [17], combining genetic algorithm and ant colony algorithm, using the strong local search ability of ACO algorithm to make up deficiencies of GA, making the efficiency of the search to improve and search results to ensure improvement is presented. On the basis of using Ford algorithm to obtain the "shortest path" of link map, implanting the path adjustment algorithm based on merged GA and ACO to adjust the path, and the search results are more ideal than a single Ford algorithm. Another approach presents an ACO to solve the shortest path problem, especially with fuzzy constraints. This algorithm consists of five sequential steps. The first step is to determine the number of possible paths from the source to the target. The second step calculates the probability of each path of possible paths. The third step calculates the expected number of ants through each path of possible paths then calculates in the fourth step the new trail of each weight component for each path of possible paths, which leads to the final step to calculate the average trail of each path.

4. Summary

This paper reviewed techniques of TE that generate optimal traffic in AS. We show that offline/online are efficient TE tools while intelligence techniques are more popular in recent years. Most researches combined intelligence to improve the performance of traffic. However, all techniques as mentioned in this paper only focus on single path optimization, source node reaches destination node only using the path which selected at the beginning. The multi-path technique improves link bandwidth when link nodes are congested. Last, but certainly not least, we will review multi-paths for TE in future research.

5. References

[1] S. Jamin, C. Jin, D. Raz, Y. Shavitt, and L. Zhang, "On the Placement of Internet Instrumentation," *INFOCOM* 2000*. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*. IEEE, pp. 295-304, 2000.

[2] P. Krishnan, D. Raz, and Y. Shavitt, "The cache location problem," *IEEE/ACM Transactions on Networking*, Vol. 8(5), pp. 568–582, 2000.

[3] X. Jia and et. al., "Optimal placement of proxies of replicated web servers in the Internet," *International Conference on Web Information Systems Engineering*, 2000.

[4] S. Jamin and et. al., "Constrained mirror placement on the Internet," *Proceedings of IEEE INFOCOM*, pp. 31–40.

[5] L. Qiu, V. Padmanabhan, and G. Voelker, "On the placement of web server replicas," *Proceedings of IEEE INFOCOM*, Vol. 3, pp. 1587–1596, 2001.

[6] R. Cohen and G. Nakibly, "A Traffic Engineering Approach for Placement and Selection of Network Services," *Proceedings of IEEE INFOCOM*, pp. 1793–1801, 2007.

[7] R. Cohen and N. Fazlollahi and D. Starobinski, "Throughput optimal on-line algorithms for advanced resource reservation in ultra high-speed networks," *arXiv*/0711.0301, *Tech. Rep*, 2007.

[8] S. Choi and Y. Shavitt, "Proxy location problems and their generalizations," *International Workshop on New Advances of Web Server and Proxy Technologies*, Providence, RI, USA, 2003.

[9] H. Jamjoom, S. Jamin, and K. Shin, "Self organizing network services," University Michigan *CSE-TR*-407-99, 1999.

[10] B. Li and et. al., "On the optimal placement of Web proxies in the Internet," *Proceedings of IEEE INFOCOM*, pp. 1282–1290, 1999.

[11] Sugih Jamin, Cheng Jin, Yixin Jin, Dan Raz, Yuval Shavitt, and Lixia Zhang, "On the placement of Internet instrumentation," *Proc. of IEEE INFOCOM*, 2000.

[13] K. Anand and N.N. Jani, "A Novel Genetic Algorithm approach for Network Design with Robust Fitness Function," *International Journal of Computer Theory and Engineering*, Vol. 2, No. 3, June, pp. 459-465, 2010.

[14] N. Gihan and A. G. Wahied, "Network Routing Protocol using Genetic Algorithms," *International Journal of Electrical & Computer Sciences IJECS-IJENS*, Vol. 10, No. 2, pp. 40–44, 2010.

[15] J. J Yin and K. S. Tang, "Resource Placement in Content Delivery Networks Using Hybrid Genetic Algorithm," *IEEE International Conference on Industrial Informatics* (*INDIN*), pp. 281–285, 2005.

[16] Xiankun Sun, Xiaoming You, and Sheng Liu, "Multi-objective Ant Colony Optimization Algorithm for Shortest Route Problem," *IEEE International Conference on Machine Vision and Human-machine Interface*, pp. 769–798, 2010.

[17] Liang Shengzhuo and Zhengru Fang, "A Path Adjustment Algorithm of The Combination of Genetic Algorithm and Ant Colony Algorithm," *IEEE International Conference on Information Science and Engineering* (*ICISE 2009*), pp. 63-65, 2009.

[18] Ammar W. Mohemmed and Nirod Chandra Sahoo,

"Particle Swarm Optimization Combined with Local Search and Velocity Re-initialization for Shortest Path Computation in Networks" *Proceedings of The* 2007 *IEEE Swarm Intelligence Symposium* (*SIS 2007*), 2007.