A Polynomial Interval Shortest-Route Algorithm for Acyclic Network

Hossain M. Akter

***Key words**: Interval; interval shortest-route problem; interval algorithm; uncertainty.*

***Abstract**. A method and algorithm is presented for solving the shortest-route problem. New algorithm is applicable to the case when the generalized length (distance, cost, time, etc.) associated with each arc is nonnegative, interval or real. An interval algorithm is developed on the base of midpoint and half-width representation of intervals and the new algorithm is more efficient than the interval algorithm that could be proposed by using traditional interval description. The complexity of the new algorithm is evaluated.*

1. Introduction

There are many reasons why network models, methods and algorithms are widely used, for instance, they exactly represent the real world systems, they facilitate extremely efficient solution to large real problems, they can solve problems with significantly more variables and constraints than can be solved by other optimization techniques, etc. [22].

Network consists of special points called *nodes* and links connecting pairs of nodes called *arc* (or *branch* or *edge* or *link*). A network is called acyclic network, if it does not have any loop. The acyclic algorithm is easier than the cyclic algorithm, because it yields fewer computations [10,25,26].

Consider a connected network $G = (N, A)$, where $N = \{1, \ldots, n\}$ is the set of the nodes and $A = \{(i, j), (k, l), \ldots, (y, z)\}$ is the finite set of arcs joining nodes in N. The cardinality of N and A are denoted by $|N|$ and $|A|$ respectively, and $|N| = n$, $|A| = m$. Let $d(i)$ is the distance label and $i \in N$; $d(i, j)$ is the generalized length (*distance*, *time*, *cost*, etc.) of arc $(i, j) \in A$; P_n is the directed route from source node s to destination node n and s, $n \in N$; the length $d(P_n)$ of the route P_n is given by $d(P_n) = \sum d(i, j)$ such that (i, j) belongs to P_n, where by convention $d(P_s) = 0$; the predecessor of node j that is denoted by $p(j)$, is started from node i of the single arc $(i, j) \in A$ in the tree terminating at j [7,8,25].

The Shortest-Route Problem (*SRP*) is concerned with determining the shortest route from an origin to a destination through a connecting network, given nonnegative distances associated with the respective arcs of the network [3,9,22,23,25].

The SRP is a classical network problem, and it is the most popular problem/model among all network problems [3,9,23]. In literature, Dijkstra algorithm [2] is considered a classical algorithm for SRP. Last five decades many variants of Dijkstra algorithm have been developed, for example, an alternative method for SRPs is proposed in [1], which reduces the upper bound of running time, and makes empirical comparisons for a certain class of networks. *Reaching*, *Pruning*, and *Buckets* are the three concepts that are used in these methods. Reaching is a label setting scheme, reaching allows a network to be pruned during computation of some of its nodes and/or branches, and bucket is a list of nodes whose labels fall within a given range.

In [2], the author assumed n nodes, and the existence of at least one route between any two nodes. Two fundamental problems were considered: to obtain the tree of minimum total length between the n nodes, and to find the route of minimum total length between two given nodes.

The interval SRP is concerned with determining the interval shortest route from an origin to a destination through a connecting network, given the interval generalized length between nodes i and j is a nonnegative, interval and interval numbers are represented by D_{ij}, $D_{ij} = [\underline{d}_{ij}, \overline{d}_{ij}]$ [4-6,11,12].

The aim of this paper is to develop simple and effective method and algorithm for solving the SRP for acyclic network under parametric uncertainties. The analysis of the complexity of the interval algorithm will be discussed.

2. Related Work

An interval algorithm is proposed for solving SRP under parametric uncertainty in [4]. The exact values of the parameters of a given network are unknown, but upper and lower limits within which the values are expected to fall are considered. The interval algorithm is developed on the base of midpoint and half-width representation of intervals. Considerable unification and simplification are obtained by using the mean-value lemma. This interval algorithm is applicable when the parameters of a given network are interval and real. The interval algorithm is applicable when the given network is acyclic. Updated version of this algorithm is presented in [6,11].

An interval algorithm for cyclic network is presented in [5,11]. Final versions of interval methods and algorithms are given for solving the well-known SRP for acyclic and cyclic networks in [12]. The formulation of the interval shortest-route algorithm for cyclic network is an interval extension of Dijktra algorithm. The author considered the interval generalized length between nodes i and j is a nonnegative, interval number. The new methods and algorithms are developed on the base of midpoint and half-width representation of intervals. These interval algorithms are applicable when the parameters are interval and real. The complexity of these algorithms is evaluated.

Both method and algorithm are more efficient than the method and algorithm that could be obtained by using traditional interval description and comparison, and the complexity of such an algorithm will be too high from the point of view of computation and practical applications.

A method to find the most reliable route in a given network is given in [25]. The probability of an arc is certain. The author converts probability to log probability. Then the shortest-route algorithm is used to find the shortest distance (log). Finally, this log probability is converted back to non-log probability. There are some limitations of this method. If the probabilities of a given network are with higher degree of uncertainty, this method can not be used to solve the problem. The author has not considered the complexity analysis of this method. To convert probability to log probability, then log probability to probability, he needs more operations. So, the complexity of the method will be higher.

Five algorithms are proposed for solving the most reliable route problem in finite fuzzy acyclic and cyclic networks in [12,13]. The uncertainty about the reliability of a route is represented in a possibilistic setting. The plausibility of not being stopped on a segment of the route is described using the corresponding possibility. The concept of interval possibility is introduced to increase the degree of uncertainty. These algorithms maximize the possibility of not being stopped on the route between an origin node and a destination node. The complexity of these algorithms is evaluated. Brief description of the algorithms is given below:

1. The first and second algorithms are based on the usage of "and" and "product" operators to determine the strongest route, that is, the most reliable route in a finite fuzzy acyclic network. The first algorithm takes less time for computations than the second algorithm. So, the first algorithm is better suited for large network.

2. The third algorithm uses multiplication of interval possibilities and yields directly the largest interval possibility of not being stopped on the route.

3. The fourth and fifth algorithms are based on the concept of interval possibility for acyclic network and cyclic network, respectively. Only once at the beginning, the transformation of the initial representation of intervals possibilities into logarithmic form is accomplished, and then the simple midpoint algorithm for solving interval acyclic algorithm and interval cyclic algorithm is applied, respectively.

A variant of SRP has considered in [17,18,19]. Consider a directed network $G = (N, A)$, where N is the set of nodes and A is the set of arcs, and $s, n \in N$. The costs (travel times) of each arc is given by an interval. Intervals represent ranges of possible costs. An interval $[\underline{d}_{ij}, \bar{d}_{ij}]$ is associated with each arc $(i, j) \in A$, and $0 \le \underline{d}_{ij} \le \bar{d}_{ij}$. A route H from source to destination is said to be a Robust Shortest Route (*RSR*) if it has the smallest (among all routes from source to destination) maximum (among all possible scenarios) robust deviation.

In [19], the authors proposed a branch and bound algorithm for the RSR problem with interval data. The new algorithm is based on a lower bound and on some reduction rules which work by exploiting some properties of the particular branching strategy. The algorithm starts by initializing the structures of r, the root of the search-tree, which is then inserted into the set of nodes to be examined. An iterative statement is then repeated until the search-tree has been completely examined. The authors tested their methods on different networks: random networks, real networks, etc.

In [17], the authors presented an exact algorithm for RSR problem with interval data. The algorithm is based on the conjecture that a RSR is one of the first routes in a shortest route ranking in a simple directed network, where the cost on each arc (i, j) is equal to $\bar{d}_{ij}$. They adopted the algorithm which is based on the concept of route deletion, and also implemented the Dijkstra algorithm to evaluation of the robustness cost of a given route. The algorithm works in the following way: a procedure ranks routes in the simple directed network. For each route retrieved, the respective robustness cost is calculated. The algorithm stops when a lower value for the robustness costs of the routes not yet examined matches an upper bound for the same routes. The limitations of the proposed algorithm are as follows: a) if the robust route from s to t is long, all the routes from s to t will tend to be long, and the shortest route algorithm will be slower, b) if the robust route from s to t is long, more alternative routes will exist between s and t, and the algorithm need more iterations to converge, c) the new method obtains poor results on problems based on Karaşan networks. The main advantage is that the algorithm gives the optimal solution of some network problems.

In [18], two versions of novel exact algorithms are given for the RSR problem with interval data, and these algorithms are based on Benders decomposition. The Benders decomposition approach is the best one for networks with low arc density, and the branch and bound method given in [19], is the most promising while the arc density increases. The authors made an experiment on real road networks that showed that the Benders decomposition approach is the most appropriate for this type of networks. Moreover, the choice of the most appropriate approach is strictly connected with the characteristics of the problem to be solved.

In [24], the authors examined a specific SRP in acyclic network, in which arc costs are unknown functions of certain environment variables at network nodes, and each of these variables evolve according to an independent

Markov process. The vehicle can wait at a node (at a cost) in anticipation of more favorable arc costs. First, the authors developed two recursive procedures for the individual arc case, based on successive approximations, and policy iteration. Several procedures have been used to determine which of the environment states at each node are *green* (the vehicle departs immediately) and which are *red* (the vehicle waits), based on successive approximations, policy iteration, and parametric linear programming methods. The complexity of this method is $O(n^2K + nK^3)$, where n is the number of nodes and K is the number of Markov states at each node.

Sometimes single objective function may not be sufficient to characterize many practical problems completely. In a real transportation network several objectives, i.e., time, cost, distance, etc. can be assigned to each arc. If only one objective is given on each arc, the solution of the problem can be obtained by classical shortest-route algorithm, given in [2]. When more than one objective is given on each arc, the solution of the problem can not be obtained by classical shortest-route algorithm. The shortest route may be not wise to use because it could be expensive. To deal with a real problem with more than one objective, new variants of classical shortest-route algorithm have been developed, which are called the *bicriterion* or *multi-criteria* shortest-route algorithms [12,14-16,27].

In [27], the authors proposed a method to solve the fuzzy SRP. The weighted additive method is introduced to solve a multiple objective integer programming problem, which met the requirements of the Network LPs constraints. Weights in the weighted additive model show the relative importance of the goals. For simplicity, the authors assumed that the importance of the four objectives is the same. Therefore, all objective functions were reformulated as a single objective function, and one need not to add the constraints of integer programming. The fuzzy shortest route was obtained when the model met the requirements of the Network LPs constraints. This new approach reduced the complexity of solving the basic fuzzy shortest route formulation. The author assumed that the importance of all objective functions is same.

Mixed Integer Linear Programming (ILP) approach is proposed in [12,14,15] to solve the bicriterion network problem. The method is based on the approach, proposed in [28], for solving multicriterion continuous problems, which introduces fuzzy sets of the values "near to the optimal values" for each criterion. Consider a network $G = (N, A)$, where $N = \{1, \ldots, n\}$ is the set of the nodes and $A = \{(i, j), (k, l), \ldots, (y, z)\}$ is finite set of directed arcs joining nodes in N. Assume we have $|A| = m$ arcs. Each arc $(i, j) \in A$ has two attributes, for example, $d_{ij} = (d'_{ij}, d''_{ij})$. d'_{ij} is the distance between node i and node j, d''_{ij} is the travel time between node i and node j.

In [15], Mixed ILP approach is proposed to solve the multicriterion network problem. The method also is based on the approach, proposed in [28], for solving multicriterion continuous problems, which introduces fuzzy sets of the values "near to the optimal values" for each criterion. Consider a network $G = (N, A)$, where $N = \{1, \ldots, n\}$ is the set of the nodes and $A = \{(i, j), (k, l), \ldots, (y, z)\}$ is finite set of directed arcs joining nodes in N. Assume we have $|A| = m$ arcs. Each arc $(i, j) \in A$ has three attributes, for example, $d_{ij} = (d'_{ij}, d''_{ij}, d'''_{ij})$. d'_{ij} is the distance between node i and node j, d''_{ij} is the travel time between node i and node j, and d'''_{ij} is the travel cost between node i and node j.

3. Theoretical Preliminaries

An interval number is a pair of real numbers ($\underline{r}$, $\bar{r}$), with $\underline{r} \le \bar{r}$. The interval analysis concepts are introduced in [20,21]. Let R be an interval. We will denote its lower (left) endpoint by $\underline{r}$ and its upper (right) endpoint by $\bar{r}$, so that $R = [\underline{r}, \bar{r}]$.

The set of all intervals will be denoted by $I(R)$. Let R, $S \in I(R)$, and let $*$ denote any of the interval arithmetic operations, $* = +, -, \times, /$. Then the set theory definition of the interval arithmetic operations is as follows:

(1) $R * S = \{r * s \mid r \in R,$

It follows that the sum of $R = [\underline{r}, \bar{r}]$, $S = [\underline{s}, \bar{s}]$ denoted by $R + S$, is the interval

$R + S = [\underline{r}, \bar{r}] + [\underline{s}, \bar{s}] = [\underline{r} + \underline{s}, \bar{r} + \bar{s}]$.

The product $R \times S$ is again an interval

$R \times S = [\min\{\underline{r}\underline{s}, \underline{r}\bar{s}, \bar{r}\underline{s}, \bar{r}\bar{s}\}, \max\{\underline{r}\underline{s}, \underline{r}\bar{s}, \bar{r}\underline{s}, \bar{r}\bar{s}\}]$.

For R, $S > 0$ the definition reduces to

(2) $R \times S = [\underline{r}\underline{s}, \bar{r}\bar{s}]$

The half-width of an inter $R = [\underline{r}, \bar{r}]$ is the real number, $w(R) = \frac{1}{2}(\bar{r} - \underline{r})$, and the midpoint of R is the real number, $m(R) = (\underline{r} + \bar{r})/2$.

Using the set inclusion relation $\subseteq$ and the relation $\le$, we can define the supremum-like (*sup*) and infimum-like (*inf*) elements:

(3) $\sup(R, S) = [\sup(\underline{r}, \underline{s}), \sup(\bar{r}, \bar{s})]$

(4) $\inf(R, S) = [\inf(\underline{r}, \underline{s}), \inf(\bar{r}, \bar{s})]$

To compare intervals the concept of metric ρ is introduced. For each R and S in $I(R)$ the distance ρ is defined by

(5) $\rho(R, S) = \frac{1}{2}\{|\underline{r} - \underline{s}| + |\bar{r} - \bar{s}|\}$

Now the intervals R and S can be compared. The following important results hold in [4].

$R \le S$ iff (if and only if)

(6) $\rho(R, \inf(R, S)) \le \rho(S, \inf(R, S))$

In a similar way,

$R \ge S$ iff

(7) $\rho(R, \sup(R, S)) \le \rho(S, \sup(R, S))$

Two intervals R and S are said to be equivalent $R \sim S$ if the following condition holds:

(8) $\rho(R, \sup(R, S)) = \rho(S, \sup(R, S))$

(9) $\rho(R, \inf(R, S)) = \rho(S, \inf(R, S))$

It means that $|\underline{r} - \underline{s}| = |\overline{s} - \overline{r}|$, i.e., the midpoints of R and S coincide.

In practical cases when $R \sim S$ and one have to make a choice in the sense of $\le$, the condition (6) should be modified. We say that $R \le S$ if

(10) $\rho(R, \inf(R, S)) = \rho(S, \inf(R, S))$ and $\underline{r} \le \underline{s}$

or

(11) $\rho(R, \inf(R, S)) = \rho(S, \inf(R, S))$ and $\overline{r} \le \overline{s}$

We use, further, the notation $R \le S$ in the usual sense, when $\underline{r} \le \underline{s}$ and $\overline{r} \le \overline{s}$, and in the case of inclusion, $R \subseteq S$, when $\rho(R,\inf(R, S)) \le \rho(S,\inf(R, S))$.

The conditions (6) and (7) lead to the following result, as proven in [4].

Let $m(P)$ denote the midpoint of P, $m(P) = (\underline{p} + \overline{p})/2$. Then

(12) $R \le S$ iff $m(R) \le m(S)$

Let $[m(R), \Delta(R)]$ denote the interval R, $R = [\underline{r}, \overline{r}]$, where $m(R) = (\underline{r} + \overline{r})/2$ is the midpoint of R, and $\Delta(R) = (\underline{r} - \overline{r})/2$ is the half-width of R, so that

(13a) $R = [m(R) - \Delta(R), m(R) + \Delta(R)]$

(13b) $R = [m(R), \Delta(R)]$

The following result is easily shown:

Let R, S, and $T \in I(R)$. Then $T = R + S$ iff

(14) $m(T) = m(R) + m(S)$

(15) $\Delta(T) = \Delta(R) + \Delta(S)$

4. Method and Algorithm

In many practical cases, the parameters of the network models are not exactly known, they are uncertain. A typical way to express these uncertainties in the edge weights is to utilize tools based on probability theory, interval mathematics, fuzzy sets theory, etc.

The aim is to find the shortest route between a *source* node 1 and any *destination* node t in a network with n nodes, $t \le n$. An interval extension of the well-known acyclic algorithm can be obtained, using the interval operation +, the metric ρ as defined in (5), and the conditions (6) or (10), (11).

Let D_{ij} and U_j denote the interval distance between nodes i and j, and the shortest interval distance from the source node (node 1) to node j, correspondingly. The destination node is node n.

The interval values of $U_j = [\underline{u}_j, \overline{u}_j]$, $j = \overline{2, n}$ may be computed recursively using the interval formula

(16) $U_j = \min_i \{U_i + D_{ij}\}$

where $U_i + D_{ij} = [\underline{u}_i + \underline{d}_{ij}, \overline{u}_i + \overline{d}_{ij}]$, and $U_1 = [0, 0]$. The operator min{} is performed on the basis of the metric (5) and the conditions (6) or (10), (11). This way an interval extension of the well-known shortest-route algorithm for acyclic network is obtained.

We present a more effective algorithm, using the midpoint and half-width notation, (13b) and the conditions (12), (14) and (15).

Let u_j denote the real shortest distance from 1 to node j. The real values u_j, $j = \overline{2, n}$ are computed using the recursive noninterval formula

(17) $u_j = \min_i \{u_i + d_{ij}\}$

where d_{ij} is the midpoint of D_{ij}, $u_1 = 0$.

To obtain the optimal solution of the SRP, it is important to identify the nodes encountered along the route and the corresponding interval widths. The following labeling of node j is used

(18) node j Label = $[u_j, k, \Delta_{kj}]$

where k is the node immediately preceding j that leads to the shortest distance u_j, and Δ_{kj} is the half-width of D_{kj}.

Further it is assumed that the network is described using interval notation with midpoint and half-width (13b). It is also assumed a natural consecutive numbering of nodes from 1 to n, such that the number of any node i, $i \in N \mid 1$ is greater than the number of any immediately preceding node k, $k \in N$, and where N is the set of nodes, and N_i is the set of all preceding node.

The generalized steps of the interval acyclic algorithm are summarized as follows:

***Step* 1**. Assign the label [0, –, 0] to the source node 1. Set $j = 1$.

***Step* 2**. Set $j = j + 1$. Compute the shortest distance from source node 1 to node j, by using recursive formula (17). Label node j by using (18).

If $j < t$ repeat *step* 2.

***Step* 3**. Obtain the optimum route H^* between nodes 1 and node n, starting from node n and tracing backward through the nodes using the label's information.

***Step* 4.** Obtain the half-width $\Delta(U_n)$ of the interval solution U_n, adding the corresponding Δ_{ij} encountered along the optimum route H^*

$$\Delta(U_n) = \sum_{(i,j)\in H^*} \Delta_{ij}.$$

***Step* 5.** Obtain the interval solution U_n, $U_n = [u_n - \Delta(U_n), u_n + \Delta(U_n)]$.

The algorithm provides the shortest route between node 1 and any node j, $j \le n$ in the network.

4.1. Analysis of the complexity of the interval shortest route algorithm for acyclic network

Consider the network in *figure 1*. The cardinality $|N_j|$ of the set of entering arcs N_j into node j is $(j-1)$, $|N_j| = j-1$. To calculate (shortest distance from 1 to node j, $j = \overline{2,n}$), we need the following addition(s) and comparison(s):

$u_2 = u_1 + d_{12}$ ⇒ We need only 1 addition to determine u_2.

$u_3 = \min\{(u_1 + d_{13}), (u_2 + d_{23})\}$ ⇒ We need only 2 additions and 1 comparison to determine u_3.

$u_4 = \min\{(u_1 + d_{14}), (u_2 + d_{24}), (u_3 + d_{34})\}$ ⇒ We need only 3 additions and 2 comparisons to determine u_4.

$u_5 = \min\{(u_1 + d_{15}), (u_2 + d_{25}), (u_3 + d_{35}), (u_4 + d_{45})\}$ ⇒ We need only 4 additions and 3 comparisons to determine u_5.

.

.

.

$u_{(n-1)} = \min\{(u_1 + d_{1(n-1)}), (u_2 + d_{2(n-1)}), (u_3 + d_{3(n-1)}), \ldots, (u_{n-2} + d_{(n-2)(n-1)})\}$ ⇒ We need only $(n-2)$ additions and $(n-3)$ comparisons to determine $u_{(n-1)}$.

$u_n = \min\{(u_1 + d_{1n}), (u_2 + d_{2n}), (u_3 + d_{3n}), \ldots, (u_{n-2} + d_{(n-2)(n-1)}), (u_{(n-1)} + d_{(n-1)n})\}$ ⇒ We need only $(n-1)$ additions and $(n-2)$ comparisons to determine .

Hence, to obtain u_j we need only $(j-1)$ additions and $(j-2)$ comparisons.

The number of additions is $\sum_{j=2}^{n}(j-1)$.

The number of comparisons is $\sum_{j=2}^{n}(j-2)$.

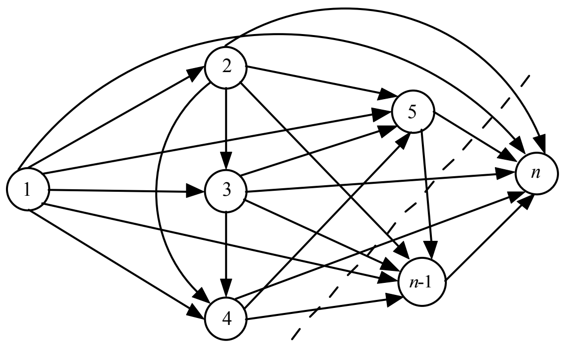

Figure 1. Acyclic network n node

We set $\chi = j-1$ and $\delta = j-2$, and we obtain:

$$\sum_{\chi=1}^{n-1}\chi = \frac{(n-1)\times n}{2}$$

$$\sum_{\delta=1}^{n-2}\delta = \frac{(n-2)\times(n-1)}{2}.$$

The total number of additions is ϑ, $\vartheta = \frac{(n-1)\times n}{2} + (n-1)$ (to get total half-width) + 2 additions to obtain the traditional interval representation.

The total number of comparisons is ε, $\varepsilon = \frac{(n-2)\times(n-1)}{2}$.

So, the running time of the algorithm is bounded by $O(addi = \vartheta, comp = \varepsilon)$.

Note that if the interval formula (16) were used, then each comparison of two intervals $V = [\underline{v}, \overline{v}]$ and $W = [\underline{w}, \overline{w}]$ includes the following comparisons and additions:

***Step* 1**. if $\overline{v} \le \underline{w}$ → set Lab = V, go to the next interval comparison ⇒ 2 comparisons;

***Step* 2**. if $\overline{w} \le \underline{v}$ → set Lab = W, go to the next interval comparison ⇒ 2 comparisons;

***Step* 3**. if $V = W$ ($\underline{v} = \underline{w}$ and $\overline{v} = \overline{w}$) → set Lab = V, W, go to the next interval comparison ⇒ 2 comparisons;

***Step* 4**. if $\underline{v} \le \underline{w}$ and $\overline{v} \le \overline{w}$ → set Lab = V, go to the next interval comparison ⇒ 4 comparisons;

***Step* 5**. if $\underline{w} \le \underline{v}$ and $\overline{w} \le \overline{v}$ → set Lab = W, go to the next interval comparison ⇒ 4 comparisons;

***Step* 6**. if $\underline{v} \le \underline{w}$ → set $\rho_1 = \rho(V, \inf) = \overline{v} - \overline{w}$; set $\rho_2 = \rho(W, \inf) = \underline{w} - \underline{v}$ ⇒ **2** comparisons, 2 additions;

if $\rho_1 < \rho_2$ → set Lab = V

else set Lab = W go to the next interval comparison ⇒ 1 comparison;

***Step* 7**. if $\underline{w} \le \underline{v}$ → set et $\rho_1 = \rho(V, \inf) = \underline{v} - \underline{w}$; set $\rho_2 = \rho(W, \inf) = \overline{w} - \overline{v}$ ⇒ 2 comparisons, 2 additions;

if $\rho_1 \le \rho_2$ → set Lab = V

else set Lab = W ⇒ 2 comparison.

To compare two intervals 1 to 21 comparisons and 0 to 4 additions are needed. So, if we develop interval shortest-route algorithm based on traditional interval representation, the complexity of the algorithm will be very high.

Comment

The analysis of the complexity is very important for two reasons: *practical reasons* and *theoretical reasons*. The first reason can be summarized as a need to obtain the execution-time that are needed in the implementation of algorithm. The second reason for the complexity analysis of the algorithm is the desirability of quantitative standards that would allow the comparison of more than one algorithm designed to solve the same problem.

The *analysis of the complexity of the interval shortest route algorithm* is to provide upper bounds on the amount of computational work (comparisons (*comp*) and additions

Table 1. The computations of all iterations for *figure 2*

Node j	Computation of u_j	Connected from	Label
1	$u_1 = 0$	-	[0, -, 0]
2	$u_2 = 0 + 8 = 8$	node 1	[8, 1, 1]
3	$u_3 = 0 + 9 = 9$	node 1	[9, 1, 1]
4	$u_4 = \min\{0 + 10, 8 + 11, 9 + 5\} = 10$	node 1	[10, 1, 1]
5	$u_5 = \min\{8 + 25, 10 + 7\} = 17$	node 4	[17, 4, 1]
6	$u_6 = \min\{9 + 15, 10 + 12\} = 22$	node 4	[22, 4, 1]
7	$u_7 = \min\{17 + 9, 22 + 8\} = 26$	node 5	[26, 5, 1]

(*addi*)) involved in the application of interval shortest route algorithm for acyclic network.

It is assumed that a comparison and an addition require approximately the same unit of time. The worst-case conditions for the execution of an algorithm means that the required number of elementary operations to terminate the algorithm is maximum [22].

4.2. Numerical Example

Consider the network in *figure 2*. The generalized length of the arcs are uncertain and given by intervals, in the form (13b).

Using the algorithm, as described in section 4, we obtain the results for nodes 1, 2, …, 7, that are put on *table 1*. The computations for all iterations are summarized directly on *figure 2*.

The optimal solution is obtained tracing backward from node 7 and using the label's information

$7 \to [26,5,1] \to 5 \to [17,4,1] \to 4 \to [10,1,1] \to 1.$

The half-width of the optimal solution is as follows:

$$\Delta_7 = \Delta_{57} + \Delta_{45} + \Delta_{14} = 1 + 1 + 1 = 3.$$

Hence, U_7 = [26 – 3, 26 + 3] = **[23, 29]**.

The algorithm provides the shortest interval distance between node 1 and any other node. In *figure 2*, the solid lines show the obtained the interval shortest route (the desired route) between the source and the destination node namely $1 \to 4 \to 5 \to 7$.

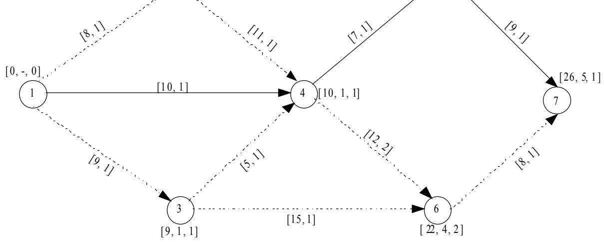

Figure 2. Acyclic network with midpoint and half-width notation

Note that if the interval formula (16) were used, at node 7, for example, we would have to compare two intervals:

[15, 19] + [8, 10] = [23, 29] and

[19, 25] + [7, 9] = [26, 34].

5. Conclusions

In this paper, we proposed a method and an algorithm for solving shortest-route problem for acyclic network based on midpoint and half-width representation of intervals (13b), and the conditions (12), (14), and (15). The new interval algorithm is applicable when the parameters are real or interval valued.

This approach yields simple and computationally effective algorithm, when the exact values of the parameters are unknown, but upper and lower limits within which the values are expected to fall are given. Instead of comparing intervals using the distance (5) and infimum-like intervals (4), the conditions (10) and (11), the algorithm compares real values, i.e., the midpoints of the intervals.

The complexity of the interval acyclic algorithm is evaluated, it is a polynomial algorithm.

Acknowledgements

The author expresses his gratitude to Prof. Dr. Geo I. Gatev for being his PhD supervisor and continuous guidance over the years.

References

1. Denardo, E. V., B. L. Fox, Shortest-Route Methods: 1. Reaching, Pruning and buckets, Operations Research, 27, 1979, 161-186.
2. Dijkstra, E. W., A Note on Two Problems in Connexion with Graphs, Journal of Numerische Mathematik, 1, 1959, 269-271.
3. Dreyfus, S. E., An appraisal of some shortest-path algorithms, Journal of Operations Research, 17, 1969, 395-412.
4. Gatev, G., Interval analysis approach and algorithms for solving network and dynamic programming problems under parametric uncertainty, Proceedings of the Technical University of Sofia, 48, 1995, 345-350.

5. Gatev, G., A. Hossain, Interval Shortest Route Algorithms, Proceedings of the International Conference, Automation & Information, Sofia, Bulgaria, 2000, 68-71.

6. Gatev, G., A. Hossain, Interval Algorithms for Solving Minimal Spanning Tree and Shortest-Route Models, Journal of Information Technologies and Control, 4, 2007, 37-46.

7. Glover, F., D. Klingman, New Sharpness Properties, Algorithms and Complexity bound for Partitioning Shortest Path Procedures, Journal of Operations Research, 37, 1989, 542-546.

8. Glover, F., D. Klingman, N. V. Phillips, R. F. Schneider, New Polynomial Shortest Path Algorithms and their Computational Attributes, Journal of Management Science, 31, 1985 1106-1128.

9. Golden, B., Shortest-Path Algorithm: A Comparison, Operations Research, 24, 1976, 1164-1168.

10. Hillier, S. F., G. J. Lieberman, Introduction to Operations Research, Fifth edition. McGraw-Hill Publishing Company, New York, 1990.

11. Hossain, A., Interval Algorithms for solving Network and Dynamic programming under parametric uncertainty, Master's Thesis, Technical University of Sofia, 1999.

12. Hossain, A., Network Models under Parametric Uncertainty, PhD Thesis, Technical University of Sofia, 2009.

13. Hossain, A., G. Gatev, Polynomial Algorithms for solving the Most Reliable Route Problem, Journal of Information Technologies and Control, Ed. K. Boyanov, 4, 2009, 17-26.

14. Hossain, A., Method and Algorithm for solving the Bicriterion Network Problem, Proceeding of the 2010 International Conference on Industrial Engineering and Operations Management, Dhaka, Bangladesh, 2010.

15. Hossain, A., Methods and Algorithms for solving the Bicriterion and Multicriterion Network Problem, Journal of International Journal of Logistics and Transportation Research, 1, 2010, 39-47.

16. Martins, E. Q. V., On a multicriteria shortest path problem, European Journal of Operational Research, 16, 1984, 236-245.

17. Montemanni, R., L. M. Gambardella, An exact algorithm for the robust shortest path problem with interval data, Computers & Operations Research, 31, 2004, 1667-1680.

18. Montemanni, R., L. M. Gambardella, The robust shortest path problem with interval data via Benders decomposition, A Quarterly Journal of Operations Research, 3, 2005, 315-328.

19. Montemanni, R., L. M. Gambardella, A. V. Donati, A branch and bound algorithm for the robust shortest path problem with interval data, Operation Research Letters, 32, 2004, 225-232.

20. Moore, R. E., Interval Analysis, Prentice Hall, Englewood Cliffs, N.J., 1966.

21. Moore, R. E., Methods and Applications of Interval Analysis, SIAM, Philadelphia, 1979.

22. Phillips, Don T., A. Garcia-Diaz, Fundamentals of Network Analysis, Prentice-Hall Inc., Englewood Cliffs, N.J., 1981.

23. Pollack, M., W. Wiebenson, Solution of the Shortest-Route Problem - A Review, Journal of Operations Research, 8, 1960, 224-230.

24. Psaraftis, H. N., J. N. Tsitsiklis, Dynamic Shortest Paths in Acyclic Networks with Markovian Arc Costs, Operations Research, 41, 1993. 91-101.

25. Taha, H. A., Operations Research, An Introduction, Eighth Edition, Pearson Education Inc., Delhi, 2007.

26. Wanger, D. K., Shortest Paths is almost acyclic graphs, Operations Research Letters, 27, 2000, 143-147.

27. Yu, J.-R., T.-H. Wei, Solving the Fuzzy Shortest Path Problem by using a Linear Multiple Objective Programming, Journal of the Chinese Institute of Industrial Engineers, 24, 2007, 360-365.

H.-J. Zimmermann, Fuzzy Set Theory - and its Applications, Second Revised Edition, Kluwer Academic Publishers, Boston, 1991.

Manuscript received on 22.11.2012

Akter Hossain *was born in Dhaka, Bangladesh. He received his academic degrees in B.Sc. in Industrial Engineering (1997), M.Sc. in Communication Engineering (1999), M.Phill. (2002) and PhD (2009). At the present, he has been working as Associate Professor & Head, Department of Computer Science & Engineering, IBAIS University, Dhaka, Bangladesh. His research and teaching interests are in the field of E-Commerce, Decision-Making, Management, Production Operation Management, Operations Research, etc. Dr. Hossain has 23 International Publications published in France, Bulgaria, India and Bangladesh.*

Contacts:
e-mail: akter_h@yahoo.com