SOLVING THE SHORTEST PATH PROBLEM WITH IMPRECISE ARC LENGTHS USING A TWO-STAGE TWO-POPULATION GENETIC ALGORITHM

FENG-TSE LIN AND TENG-SAN SHIH

Department of Applied Mathematics
Chinese Culture University
No. 55, Hwa-Kang Road, Yangminshan, Taipei 111, Taiwan
ftlin@faculty.pccu.edu.tw

Received August 2010; revised January 2011

ABSTRACT. This study investigates how to solve the shortest path problem with imprecise arc lengths using a two-stage two-population genetic algorithm (GA). This approach can conveniently represent imprecise numerical quantities, and therefore, it is able to handle imprecise arc lengths. In its first stage, the proposed GA simulates a fuzzy number by partitioning an imprecise arc length into a finite number of subintervals. Each subinterval represents a partition point. A random real number in [0, 1] is first assigned to each partition point. The GA then evolves the values in each partition point, with the final values in each partition point representing the membership grades of that fuzzy number. Thus, it is possible to obtain estimated values for the originally imprecise arc lengths, and the fuzzy problem becomes a defuzzified instance. The second stage of the GA is to search for the best solution to the defuzzified instance using a scheme in which two candidate populations evolve simultaneously. The first population comprises a set of feasible candidate solutions, and the second population consists of infeasible candidate solutions. The two solution populations are separately maintained and evolved, but their offspring may flow from one population into the other. Experimental results show that the proposed two-stage two-population GA approach obtains better results than other fuzzy shortest path approaches.

Keywords: Fuzzy number, Fuzzy shortest path problem, Imprecise arc lengths, Two-stage two-population genetic algorithm, Signed-distance ranking method

1. Introduction. A fundamental problem in the area of graph theory is that of finding the shortest path in a given graph. Over the past few years, the shortest path problem has often been posed as a subset of other optimization problems [12]. The analysis of fuzzy counterparts of the shortest path problem has also drawn much recent attention [18]. The main advantage, compared with the nonfuzzy problem formulation, is that the decision-maker is not forced into a precise formulation. Fuzziness can be introduced into a network in a variety of ways, for example, through edge capacities, vertex restrictions or arc lengths [1,2,5,10,14].

Dubois and Prade [6] first introduced the fuzzy shortest path problem in 1980. The major drawback of their formation is its lack of interpretation. More specifically, a fuzzy shortest path may be found that does not correspond to an actual path in the network. Klein [10] presented new models based on fuzzy shortest paths that circumvent this problem. He developed a hybrid multi-criteria algorithm based on fuzzy dynamic programming. Klein analyzed the fuzzy shortest path algorithm in terms of general fuzzy mathematical programming, but his proposed approach, in which he assumed that each of the arc lengths must fall between one and a fixed integer, was not reasonable. Mares
and Horak [18] proposed that the uncertainty connected with the input data for a network can be described and investigated by using fuzzy sets and fuzzy quantities theory. They showed that the uncertainty connected with more complex paths and reserves can be derived after summing fuzzy quantities.

Chanas and Kolodziejczyk [2] analyzed the maximum flow in a network that permits an excess over the previously fixed quota of arc capacity. This problem is represented as a partially fuzzy linear programming task. They then presented an algorithm for searching maximum flow assuming integer values for flows on network arcs. Okada and Soper [21] developed an order relation between fuzzy numbers, based on fuzzy minima, which follows a nondominated path or Pareto optimal path from the specified node to all other nodes. They then proposed an algorithm for solving fuzzy shortest path problems on the basis of the multiple labeling method for a multi-criteria shortest path problem. Kung and Chuang [11] proposed an algorithm to solve the fuzzy shortest path problem by combining a fuzzy shortest-path-length procedure and a fuzzy similarity measure. The first procedure finds the fuzzy shortest length, and the second measures the degree of similarity between sets of fuzzy length.

Ji, Iwamura and Shao [8] proposed three concepts of fuzzy shortest path: expected shortest path, α-shortest path and the most shortest-path, and formulated three models for the fuzzy shortest path according to different decision criteria. Nayem and Pal [20] presented an algorithm based on Dijkstra’s algorithm [12] to find the fuzzy shortest paths for a network with its arc lengths as interval numbers or triangular fuzzy numbers. Sengupta and Pal [22] presented an algorithm for the shortest path problem that considers the fuzzy preference interval ordering from the points of view of both pessimistic and optimistic decision makers.

This study proposes a novel GA approach for solving the shortest path problem with imprecise arc lengths. Each imprecise arc length in the network is originally represented as a fuzzy set. The proposed GA has two stages. The first stage simulates a fuzzy number by dividing an imprecise arc length into certain partition points (i.e., subintervals). An initial random real number in [0, 1] is first assigned to each partition point. The GA then evolves the values in each partition point. The final values in each partition point represent the membership grade of that fuzzy number, and are adopted to obtain the estimated values of each imprecise arc length. Thus, the fuzzy problem becomes a defuzzified instance. The second stage uses a two-population scheme to derive the shortest path length from the source to the destination in the defuzzified instance. One population consists of feasible solutions, and the other consists of infeasible solutions. Feasible solutions are used to improve their objective path lengths, and thus obtain better solutions. Infeasible solutions are selected to reduce their constraint violations and, if possible, create new feasible solutions.

The merits of using a two-population scheme are discussed in the literature and the arguments may be briefly summarized as follows: Yuchi and Kim [25] reported success with a two-population scheme in which a specified fraction of the infeasible solutions is probabilistically accepted for breeding each generation, based on their objective function values. Kimbrough et al. [9] proposed a two-population GA for solving constrained optimization problems. They concluded that both populations evolved optimal solutions, and that beneficial genetic materials could originate in either population. These benefits typically diffuse into both populations, and the infeasible population exhibits tradeoffs between optimization vs. constraint. In our experiments, the two-population GA performed very well on the shortest path problem with imprecise arc lengths. Consequently, the fuzzy concept of the proposed two-stage two-population GA approach is different, but it provides better solutions than other fuzzy shortest-path approaches.
2. Formulation of the Problem. The shortest path problem is described as follows. A directed network is an acyclic directed graph $G = (V, E)$ with a weight function $f : E \rightarrow R$ mapping arcs to real-valued numbers. The set $V$ denotes the vertices of the graph, and the set $E$ refers to the arcs of the graph. Each directed arc $\langle i, j \rangle$ has an associated cost $c_{ij}$. The costs are specified by the cost matrix $C = [c_{ij}]$, where $c_{ij}$ denotes the arc length from vertex $i$ to vertex $j$. Any two vertices are assumed to have only one directed arc between them. The length of a path from vertex $i$ to vertex $k$, $p = \langle i, i_1, i_2, \ldots, k \rangle$, is the sum of the lengths of its constituent arcs. The goal of this problem is to find a shortest path length from a given source vertex $s \in V$ to a given destination vertex $n \in V$ in a network.

A dynamic programming formulation for the shortest path problem, based on Bellman’s equation [10], is presented as follows. A network with an acyclic directed graph $G = (V, E)$ with $n$ vertices numbered from 1 to $n$, such that 1 is the source, and $n$ is the destination is described by

$$
\begin{align*}
    f(n) &= 0 \\
    f(i) &= \min_{j<i} \{c_{ij} + f(j) | \langle i, j \rangle \in E \}.
\end{align*}
$$

Here, $c_{ij}$ is the arc length of the directed edge $\langle i, j \rangle$, and $f(i)$ is the length of the shortest path from vertex $i$ to vertex $n$. Bellman’s principle of optimality means that this recursion is very flexible, and constitutes the basis of the model presented below.

Traditionally, the arc lengths have often been considered as exact values, and thus the outputs for the shortest path problem are deterministic. However, in real-world applications, the arc lengths can represent transportation time or cost in traveling. Consequently, a decision-maker cannot practically represent an arc length in a network using a single precise number. The arc length can only be estimated within a certain interval. Because of this interval estimation feature, the imprecise arc lengths can be realistically and naturally represented as fuzzy sets. The fuzzy linguistics that describes imprecise arc lengths require fuzzy numbers. The prerequisites for handling fuzzy numbers are described below, and in further detail in the literature [7,15,16,23].

**Definition 2.1.** The level $\lambda$ triangular fuzzy number $\tilde{A}$, $0 < \lambda \leq 1$, denoted by $\tilde{A} = (a, b, c; \lambda)$, is a fuzzy set defined on $R$ with the membership function defined as

$$
\mu_{\tilde{A}}(x) = \begin{cases} 
    \frac{\lambda(x - a)}{b - a}, & a \leq x \leq b \\
    \frac{\lambda(c - x)}{c - b}, & b \leq x \leq c \\
    0, & \text{otherwise}
\end{cases}
$$

**Definition 2.2.** The family of all level $\lambda$ fuzzy numbers is denoted by $F_N(\lambda) = \{(a, b, c; \lambda) | \forall a < b < c, a, b, c \in R\}, 0 < \lambda \leq 1$. Let $(a, b, c; 1)$ be the triangular fuzzy number denoted by $(a, b, c)$.

**Definition 2.3.** The distance on $R$ is defined by $d(b, 0) = b$, where $b$ and $0 \in R$.

**Definition 2.4.** The ranking of $\tilde{A} = (a, b, c; \lambda)$, $\lambda \in (0, 1]$ is the signed-distance measured from $\tilde{0}_1$ and is defined by $d(\tilde{A}, \tilde{0}_1) = \frac{1}{2}(2b + a + c)$.

**Definition 2.5.** Let $\tilde{A} = (a, b, c; \lambda)$ and $\tilde{B} = (p, q, r; \lambda) \in F_N(\lambda)$. The binary operation is defined by $d(\tilde{A} \oplus \tilde{B}, \tilde{0}_1) = d(\tilde{A}, \tilde{0}_1) + d(\tilde{B}, \tilde{0}_1)$. 
we obtain

The following inequalities is derived from (9), where at least one equals sign holds. Adding one quarter of (6) to (5), one obtains

where at least one equals sign holds for all possible paths, \( i \). Rewriting the above statements as

\( f(i) = \min \{c_{ik1} + c_{ik2} + \ldots + c_{k_p(k)n} \} \) for all paths \( p = \langle i, k_1, k_2, \ldots, k_p(k), n \rangle \).

The decision-maker should choose appropriate values for parameters to satisfy

where at least one equals sign holds. Adding one quarter of (6) to (5), one obtains

where at least one equals sign holds. We can see that (7) is equivalent to

\( d \left( \hat{c}_{i1} + \hat{c}_{i2} + \ldots + \hat{c}_{m(i)n} \right) \leq d \left( \hat{c}_{ik1} + \hat{c}_{ik2} + \ldots + \hat{c}_{k_p(k)n} \right) \),

where at least one \( \approx \) holds for all possible paths from vertex \( i \) to vertex \( n \). Clearly, (8) is obtained from fuzzifying (5) and taking (6) as a fuzzified condition. Note that the relations \( \prec \) and \( \approx \) in (8) are the rankings defined in \( F_N(1) \) (see Definition 2.6). Thus, we obtain

\( d \left( \hat{c}_{i1} + \hat{c}_{i2} + \ldots + \hat{c}_{m(i)n}, \hat{0}_1 \right) = d \left( \hat{c}_{ik1}, \hat{0}_1 \right) + d \left( \hat{c}_{i1} + \hat{c}_{i2} + \ldots + \hat{c}_{m(i)n}, \hat{0}_1 \right) = c_{ik1}^* + c_{k1k2}^* + \ldots + c_{k_p(k)n}^* \)

Similarly, we have

\( d \left( \hat{c}_{ik1} + \hat{c}_{k1k2} + \ldots + \hat{c}_{k_p(k)n}, \hat{0}_1 \right) = c_{ik1}^* + c_{k1k2}^* + \ldots + c_{k_p(k)n}^* \)

The following inequalities is derived from (9),

\( c_{i1}^* + c_{i2}^* + \ldots + c_{m(i)n}^* \leq c_{ik1}^* + c_{k1k2}^* + \ldots + c_{k_p(k)n}^* \)
where at least one equals sign holds for all possible paths from vertex \(i\) to vertex \(n\). Let \(f^*(i)\) be the length of the shortest path from vertex \(i\) to vertex \(n\) in network \(G = (V, E)\) with \(\{c_{ij}^*|\langle i, j \rangle \in E\}\). From (10), we obtain \(f^*(i) = c_{ii}^* + c_{i1}^* + \ldots + c_{im(i)}^*\). Similarly, we obtain

\[
f^*(j) = c_{jj}^* + c_{j1}^* + \ldots + c_{jm(j)}^*.
\]  

Rewriting (1) as follows: for any fixed \(i\), \(f(i) \leq c_{ij} + f(j)\), \(\forall i < j\), \(\langle i, j \rangle \in E\), where at least one equals sign holds. Then,

\[
c_{ii} + c_{i1} + \ldots + c_{im(i)} \leq c_{ij} + c_{j1} + c_{j2} + \ldots + c_{jm(j)}, \quad \forall i < j, \langle i, j \rangle \in E,
\]

where at least one equals sign holds. The decision-maker should choose appropriate values for parameters to satisfy

\[
\Delta_{ii} + \Delta_{i1} + \ldots + \Delta_{im(i)} \leq \Delta_{ij} + \Delta_{jj} + \Delta_{j1} + \ldots + \Delta_{jm(j)}, \quad \forall i < j, \langle i, j \rangle \in E,
\]

where at least one equals sign holds. From (12) and (13), we obtain

\[
\tilde{c}_{ii} + \tilde{c}_{i1} + \ldots + \tilde{c}_{im(i)} \leq \tilde{c}_{ij} + \tilde{c}_{j1} + \tilde{c}_{j2} + \ldots + \tilde{c}_{jm(j)}, \quad \forall i < j, \langle i, j \rangle \in E,
\]

where at least one \(\approx\) holds. Thus, we have

\[
c_{ii}^* + c_{i1}^* + \ldots + c_{im(i)}^* \leq c_{ij}^* + c_{j1}^* + c_{j2}^* + \ldots + c_{jm(j)}^*, \quad \forall i < j, \langle i, j \rangle \in E,
\]

where at least one equals sign holds.

In summary, consider a fuzzy network \(\tilde{G} = (V, E)\) with \(n\) vertices numbered from 1 to \(n\) with imprecise arc lengths, \(\{c_{ij}|\langle i, j \rangle \in E\}\). The imprecise arc length is represented as triangular fuzzy number defined in (3). An estimate of the imprecise arc length is given by \(c_{ij}^* = c_{ij} + \frac{1}{4}(\Delta_{ij} - \Delta_{ij}) = c_{ij} + \frac{1}{4}\Delta_{ij}\), where \(\Delta_{ij1}\) and \(\Delta_{ij2}\) are parameters whose values are determined by the decision-maker so as to satisfy (13). Thus, one create a set of estimated arc lengths, \(\{c_{ij}^*|\langle i, j \rangle \in E\}\), for the fuzzy network \(\tilde{G}\). The defuzzification of the dynamic programming recursion of fuzzy shortest path problem is given by

\[
f^*(i) = \min_{i < j}\{c_{ij}^* + f^*(j)|\langle i, j \rangle \in E\}, \quad \text{and}
\]

\[
f^*(n) = 0
\]

where \(f^*(i)\) is the estimated shortest path length from vertex \(i\) to vertex \(n\) in the fuzzy network.

3. The Proposed Two-stage Two-population GA. Genetic algorithms were introduced as a computational analogy to adaptive systems. They are stochastic search techniques based on the principles and mechanisms of natural genetics, and are modeled on the principles of evolution via natural selection employing a population of individuals [16]. Investigators have applied GAs to the shortest path routing problem [3], group technology techniques based on the principles and mechanisms of natural genetics, and are modeled on the principles of evolution via natural selection employing a population of individuals [16]. This section describes a two-stage two-population GA for solving the shortest path problem with imprecise arc lengths. The proposed approach utilizes a different fuzzy concept from that described in Section 2.

3.1. The first stage. The first stage produces a defuzzified instance of the shortest path problem with imprecise arc lengths. The basic concept is described as follows: First, a fuzzy number is simulated by partitioning an imprecise arc length into a finite number of subintervals. Each subinterval represents a partition point. A random real number in \([0, 1]\) is assigned to each partition point. The GA then evolves the values in each partition point. The final values represent the membership grades of that fuzzy number. Assume that \(\tilde{w}\) is a triangular fuzzy number defined in the interval \([w - \Delta_1, w + \Delta_2]\). This interval
is further equally divided into \( t \) partitions. Let \( p_i = w - \Delta_1 + i \times (\Delta_1 + \Delta_2)/t, \( i = 0, 1, \ldots, t \) be the partition points, and let \( \hat{W}(p_i) = \mu_i \in [0, 1], \( i = 0, 1, \ldots, t, \) be the membership grade of \( p_i \) in an arbitrary fuzzy set \( W \). Thus, a discrete fuzzy set \( \hat{W} = (\mu_0, \mu_1, \ldots, \mu_t) \) is obtained, where each \( \mu_i, i = 0, 1, \ldots, t, \) is a random number in \([0, 1]\). We wish to find an estimated value of \( w \) in \([w - \Delta_1, w + \Delta_2]\) using the GA. The value \( w^* \) is estimated after computing the centroid of the fuzzy number \( \hat{w} \), which is defined on the discrete fuzzy set \( \hat{W} = (\mu_0, \mu_1, \ldots, \mu_t) \),

\[
w^* = \frac{\sum_{i=0}^{t} p_i \times \mu_i}{\sum_{i=0}^{t} \mu_i}.
\] (17)

Consider a given function \( F(x) = y \). Each \( F(x_i), i = 0, 1, \ldots, t, \) is different, and the fuzzy function is defined as

\[
F(\bar{X}) = F(\mu_0, \mu_1, \ldots, \mu_t) = \frac{\mu_0}{F(x_0)} + \frac{\mu_1}{F(x_1)} + \ldots + \frac{\mu_t}{F(x_t)},
\] (18)

while the centroid is given by

\[
\theta (F(\bar{X})) = \frac{\sum_{i=0}^{t} F(x_i)\mu_i}{\sum_{i=0}^{t} \mu_i}.
\] (19)

The estimated arc length \( c_{ij}^* \) is obtained from (18) and (19), by computing the centroid of each imprecise arc length \( \hat{c}_{ij} \).

The procedure for computing the first stage is described as follows: An initial population of size \( n \) is randomly generated from \([0, 1]^{t+1}\) according to the uniform distribution in the closed interval \([0, 1]\). Let the population be given by

\[
\hat{W}_h = (\mu_{h_0}, \mu_{h_1}, \ldots, \mu_{h_t}) = \frac{\mu_{h_0}}{p_0} + \frac{\mu_{h_1}}{p_1} + \ldots + \frac{\mu_{h_t}}{p_t},
\]

where \( h = 1, 2, \ldots, n; \mu_{h_i} \) is a real number in \([0, 1]\) and \( p_i \) is a partition point in the given interval with \( i = 0, 1, 2, \ldots, t \). Each individual \( \hat{W}_h, h = 1, 2, \ldots, n, \) in the population is a chromosome. Each chromosome \( \hat{W}_h, h = 1, 2, \ldots, n, \) in the population is evaluated using (19) to create the estimated value of each imprecise arc length. The fitness value of each chromosome is then obtained from (17). The chromosomes in the population can be rated in terms of their fitness values. The total fitness value of the population is given by \( T \). The cumulative fitness value (a partial sum) is calculated for each chromosome \( S_h, h = 1, 2, \ldots, n. \) The intervals, \( I_1 = [0, S_1], I_j = [S_{j-1}, S_j], j = 2, 3, \ldots, n - 1, \) and \( I_n = [S_{n-1}, S_n] \) are constructed for the selection process. The roulette wheel mechanism [19] was chosen as the selection strategy. The selection process begins by spinning the roulette wheel \( n \) times. A random number \( r \) from the range \([0, T]\) is generated each time the wheel is spun. If \( r \in I_1 \), then chromosome \( \hat{W}_1 \) is selected; otherwise, the \( k \)th chromosome \( \hat{W}_k \), \( 2 \leq k \leq n, \) is chosen if \( r \in I_k \). This selection process is continued until the next generation has been created. Finally, the new population is renamed as \( \hat{Y}_1, \hat{Y}_2, \hat{Y}_3, \ldots, \) in the order they were picked. This procedure tends to choose \( \hat{W}_h, h = 1, 2, \ldots, n, \) with higher fitness to go on into the next generation.

Crossover is the most powerful operation in GAs, and works by recombining the genetic material of two parent chromosomes to produce offspring for the next generation. The crossover used in the first stage is the simple one-point method, which randomly selects one cut-point and exchanges the right-hand parts of two parents to generate offspring [17].
Mutation is a background operation that produces random changes in various chromosomes. Mutation resets one randomly selected position to a randomly generate number in the range of 1 to \( n \), where \( n \) is the number of nodes. The number of selected positions for mutation in the population is related to the mutation rate.

Finally, the GA program is terminated after a number of generations have been produced. Let the last population be \( W_1^*, W_2^*, \ldots, W_n^* \). The maximum fitness value is the best chromosome in the population, and the best chromosome represents the best solution to the problem. Let the best chromosome be \( W_h^* = (\mu_{h0}^*, \mu_{h1}^*, \ldots, \mu_{ht}^*) = \frac{\mu_{h0}^*}{p_0} + \frac{\mu_{h1}^*}{p_1} + \ldots + \frac{\mu_{ht}^*}{p_t} \), \( 1 \leq h \leq n \). Thus, the estimated value of each fuzzy arc length \( c_{ij} \) defined in (2) is calculated as

\[
 c_{ij}^* = \frac{\sum_{i=0}^{t} p_i \times \mu_{hi}^*}{\sum_{i=0}^{t} \mu_{hi}^*}
\]

where \( p_i = c_{ij} - \Delta_1 + i \times (\Delta_1 + \Delta_2)/t, \ i = 0, 1, \ldots, t, \) is defined on the interval \([c_{ij} - \Delta_1, c_{ij} + \Delta_2]\).

3.2. **The second stage.** The second stage solves the defuzzified instance produced by the first stage to obtain the best solution to the imprecise arc length problem. In this stage, a two-population scheme is adopted to find the shortest path in the given instance. The first population contains feasible solutions, and the second population contains infeasible solutions. An infeasible solution signifies a non-existing path in the network. These two populations are generated at the initialization of the algorithm and maintained throughout the execution of the program. The GA checks every individual for its feasibility then places it in the first or the second population. The structure of a representative chromosome at this stage consists of sequences of positive integers representing the numbers of vertices through which a routing path passes. The first integer locus designates the source. The chromosome length is equal to the number of vertices in the network. A path starts from the source and continues through to the destination. A chromosome represents a decoded path by listing the numbers of vertices from the source to the destination. Figure 1 shows an example of a chromosome’s numerical representation. Assume that vertex 1 is the source and vertex 8 is the destination. Starting from locus 1 (the source), the chromosome encodes the path as 1 \( \rightarrow \) 2 \( \rightarrow \) 5 \( \rightarrow \) 8 until it reaches its destination.

![Figure 1. An example of chromosome representation](image)

A feasible solution is a legal path in which any pair of consecutive vertices corresponds to a directed arc in the network. Conversely, an infeasible solution is an illegal path in which at least one pair of consecutive vertices has no corresponding directed arc. (See examples in Section 4.)

Genetic operators are applied to these two populations separately. Feasible solutions in the first population are selected to create better offspring. Infeasible solutions in the second population are selected to create feasible solutions. Of course, feasible parents may produce infeasible offspring, and infeasible parents may still produce infeasible offspring. But occasionally, infeasible parents may generate a new feasible solution that would never have been included in the first population. Both population sizes are fixed at each generation. The feasible population converges as its variance decreases, while the infeasible population never converges.
The proposed mechanism uses the strategy of replacing the worst individual in the next feasible generation by the best individual in the current feasible generation. That is, the best individual is always placed in the next generation. This strategy guarantees the best individual is included in each generation, and is generally called elitism [13,19]. The rest of the population is chosen based on the traditional (i.e., the roulette wheel) selection method. Elitism can rapidly increase the performance of GA, because it prevents the best solution found to date from being lost. The crossover operator used at this stage is the most difficult and critical task for developing a GA to solve the shortest path problem. Therefore, the following crossover operation is devised to solve this problem. Figure 1 illustrates the chromosome structure.

**Step 1.** Select two parents, parent1[1: n] and parent2[1: n], from the population using the roulette wheel selection method. Initially, let child1[1: n] := child2[1: n] := 0.


**Step 4.** For each child1[i] = 0, 1 ≤ i ≤ n, perform the following steps:

\[
\begin{align*}
  & k := \text{parent2}[i]; \\
  & \text{while (mat1}[k] \neq 0) \ k := \text{mat1}[k]; \\
  & \text{child1}[i] := k;
\end{align*}
\]

**Step 5.** For each child2[i] = 0, 1 ≤ i ≤ n, perform the following steps:

\[
\begin{align*}
  & k := \text{parent1}[i]; \\
  & \text{while (mat2}[k] \neq 0) \ k := \text{mat2}[k]; \\
  & \text{child2}[i] := k.
\end{align*}
\]

The following example shows the working of the proposed crossover.

**Step 1:** parent1 = [4 5 2 6 1 7 8 3], parent2 = [3 6 5 2 8 7 4 1]
child1 = [0 0 0 0 0 0 0 0], child2 = [0 0 0 0 0 0 0 0]

**Step 2:** p1 = 3
parent1 = [4 5 2 6 1 7 8 3], parent2 = [3 6 5 2 8 7 4 1]
child1 = [4 5 2 0 0 0 0 0], child2 = [3 6 5 0 0 0 0 0]

**Step 3:** mating1 = [0 5 0 3 6 0 0 0], mating2 = [0 0 4 0 2 5 0 0]

**Step 4:** child1 = [4 5 2 6 8 7 3 1], child2 = [3 6 5 2 1 7 8 4]

In this implementation, the mutation operator resets a selected position (mutation point) in a chromosome to a randomly generated number in the range of 1 to \( n \), where \( n \) is the number of nodes. An example of the mutation procedure follows. Assume that parent1 is [4 5 2 6 1 7 8 3], representing the path 1 → 4 → 6 → 7 → 8 and the mutation point is set at 4. After mutation, parent1 becomes [4 5 2 7 1 7 8 3], which represents another path 1 → 4 → 7 → 8.

4. **Experimental Results.** Two experimental examples were employed below to illustrate the effectiveness of the proposed two-stage two-population GA in solving the shortest path problem with imprecise arc lengths.

**Example 4.1.** Consider a network as shown in Figure 2, where the numbers on each directed arc represent the arc lengths.

From Figure 2, \( V = \{j | j = 1, 2, \ldots, 8\} \), \( E = \{(1, 2), (1, 3), (1, 4), \ldots, (7, 8)\} \), and \( c_{12} = 3 \), \( c_{13} = 2 \), \( c_{14} = 4 \), \ldots, \( c_{78} = 4 \). From (1), the shortest path starting from vertex 1 to vertex
8 obtained from \( f(1) = c_{12} + f(2) = c_{12} + c_{25} + f(5) = c_{12} + c_{25} + c_{58} + f(8) = c_{12} + c_{25} + c_{58} \) is 1 \( \rightarrow \) 2 \( \rightarrow \) 5 \( \rightarrow \) 8 with the total length 13. Next, consider the imprecise arc length problem. Assume that the decision-maker chooses the parameter values as follows: \( \Delta_{12} = 0.5, \Delta_{13} = 1, \Delta_{14} = 0.8, \Delta_{23} = 0.9, \Delta_{25} = 1, \Delta_{35} = 1.2, \Delta_{46} = 1.3, \Delta_{56} = 1.5, \Delta_{58} = 0.8, \Delta_{67} = 1 \) and \( \Delta_{78} = 0.9 \). All imprecise arc lengths can then be represented using triangular fuzzy numbers, which are given by: \( \tilde{c}_{12} = (3 - 0.2, 3.3 + 0.7), \tilde{c}_{13} = (2 - 0.5, 2.2 + 1.5), \tilde{c}_{14} = (4 - 0.2, 4.4 + 1), \tilde{c}_{23} = (1 - 0.3, 1.1 + 1.2), \tilde{c}_{25} = (4 - 1, 4.4 + 2), \tilde{c}_{35} = (6 - 0.3, 6.6 + 1.5), \tilde{c}_{46} = (4 - 0.2, 4.4 + 1.5), \tilde{c}_{56} = (1 - 0.5, 1.1 + 2), \tilde{c}_{58} = (6 - 0.3, 6.6 + 1.1), \tilde{c}_{67} = (2 - 1, 2.2 + 2) \) and \( \tilde{c}_{78} = (4 - 0.2, 4.4 + 1.1) \). Consequently, the network in Figure 2 becomes a fuzzy network with imprecise arc lengths, \( \tilde{G} = (V, E) \) where \( \{ \tilde{c}_{ij}(i, j) \in E \} \). From (4), the following estimated arc lengths are obtained for \( \tilde{G} : c_{12}^* = 3.125, c_{13}^* = 2.25, c_{14}^* = 4.2, c_{23}^* = 1.225, c_{25}^* = 4.25, c_{35}^* = 6.3, c_{46}^* = 4.325, c_{56}^* = 1.375, c_{58}^* = 6.2, c_{67}^* = 2.25 \) and \( c_{78}^* = 4.225 \). Consequently, the fuzzy network with imprecise arc lengths, \( G = (V, E) \) where \( \{ c_{ij}^*(i, j) \in E \} \), is defuzzied to a crisp network with estimated arc lengths, \( G^* = (V, E) \) where \( \{ c_{ij}^*(i, j) \in E \} \).

According to (16), the shortest path in the defuzzied network given by \( f^*(1) = c_{12}^* + f(2) = c_{12}^* + c_{25}^* + f^*(5) = c_{12}^* + c_{25}^* + c_{58}^* + f^*(8) = c_{12}^* + c_{25}^* + c_{58}^* \) is 1, 2, 5, 8, with a total length of 13.575. Here, the defuzzified shortest path is longer than the crisp shortest path by \( (f^*(1) - f(1))/f(1) \times 100 = 4.42\% \).

The proposed two-stage two-population GA approach for solving the shortest path problem with imprecise arc lengths is now implemented as described above. First, each imprecise arc length is represented by a fuzzy number. The first stage simulates fuzzy numbers by distributing the given imprecise arc length into certain partition points. The GA is then employed to evolve the values in each partition point. The final values in each partition point after several generations represent the membership grade of that fuzzy number. Let \( t = 10 \). The partition points are defined by \( p_k = c_{ij} - \Delta_{12} + t \times (\Delta_{12} + \Delta_{12})/10, t = 0, 1, \ldots, 10 \). This means that each imprecise arc length \( c_{ij} \) is divided into 11 partition points for simulating fuzzy number \( \tilde{c}_{ij} = (c_{ij} - \Delta_{12}, c_{ij}, c_{ij} + \Delta_{12}) \). The following parameters were used in the GA: (1) the population size was 100, (2) the probability of a crossover was 0.9, (3) the probability of a mutation was set to 0.003, and (4) the number of generations was 5000. The first run of the program yielded \( \mu_0 = 0.07, \mu_1 = 0.22, \mu_2 = 0.53, \mu_3 = 0.70, \mu_4 = 0.88, \mu_5 = 0.98, \mu_6 = 0.87, \mu_7 = 0.53, \mu_8 = 0.31, \mu_9 = 0.14 \) and \( \mu_{10} = 0.02 \), to represent the membership grade of the imprecise arc lengths. The estimated arc lengths were calculated according to (20) as follows: \( c_{12} = 3.0649, c_{13} = 2.0886, c_{14} = 4.1532, c_{23} = 0.8471, c_{25} = 3.8830, c_{35} = 6.2298, c_{46} = 4.3003, c_{56} = 1.2358, c_{58} = 6.1121, c_{67} = 1.8830 \) and \( c_{78} = 4.1826 \). Finally, the first stage produced a defuzzified instance of the fuzzy shortest path problem defined in (16).

In the second stage, the defuzzified instance was solved to find the best solution to the problem. The best path length obtained from the first run was 13.06, and can be compared with the result obtained with the crisp shortest path length as \( (13.06 - 13)/13 \).
= 0.4617%. The comparison of the result obtained here with that of the fuzzy approach is given as \((13.06 - 13.575)/13.575 = -3.7935\%\). Figure 3 shows another 20 runs of the same GA program with 20 different best lengths. The averaged value over these 20 best lengths was 12.87716, which is clearly better than the 13.575 obtained by the fuzzy approach.

![Figure 3. The best lengths obtained from 20 runs are all less than 13.575](image)

Consider the two-population scheme used in the second stage. The first population consists of feasible solutions, and the second population consists of infeasible solutions. Feasible solutions are used to improve their objective path lengths for obtaining better solutions. Infeasible solutions are selected to reduce their constraint violations and, if possible, to create a new feasible solution. For instance, some the feasible solutions in the first population were \([2 \ 5 \ 3 \ 6 \ 8 \ 7 \ 1 \ 4]\), \([4 \ 5 \ 2 \ 6 \ 1 \ 7 \ 8 \ 3]\) and \([4 \ 3 \ 1 \ 6 \ 5 \ 7 \ 8 \ 2]\), and their total lengths were 12.659, 13.975 and 16.688, respectively. Some infeasible solutions in the second population were \([6 \ 4 \ 3 \ 8 \ 2 \ 7 \ 5 \ 1]\), \([5 \ 8 \ 4 \ 2 \ 3 \ 6 \ 1 \ 7]\) and \([8 \ 1 \ 6 \ 2 \ 3 \ 5 \ 4 \ 7]\). Figure 4 illustrates the average length in the first population varied at each generation and also shows the convergence of the first population. Experimental results show that the first population always converged to a certain value after a number of generations, while the second population never converged.

![Figure 4. The first population converges after a certain number of generations](image)

In the first population, the number of infeasible offspring generated from feasible parents increased as the number of generations increased. The experiments started with 5000 generations, with an increment of 500 in each run, up to 6000 generations. In each run, the GA program was executed 30 times to obtain 30 feasible-to-infeasible counts. The average value was then calculated from these counts. Accordingly, the average feasible-to-infeasible counts obtained for each number of generations was 306.667, 466.733, 491.867, 634.300, 758.133, 848.433, 901.133, 992.200, 1070.233, 1204.667, 1207.400 and 1322.733, respectively, as shown in Figure 5. The mean of these 12 average feasible-to-infeasible
counts was 850.96. Conversely, the count of feasible offspring generated from infeasible parents in the second population was quite small. The average infeasible-to-feasible counts obtained for each different number of generations was 4.667, 3.233, 4.333, 3.567, 3.933, 4.833, 5.167, 4.033, 5.100, 3.633, 4.267 and 4.367, respectively, as shown in Figure 6. The mean of these 12 average infeasible-to-feasible counts was 4.26. Although the mean of infeasible-to-feasible counts was relatively small, it helps the GA to find better solutions during the evolution processes.

**Example 4.2.** Consider another network shown in Figure 7, where the arc lengths are presented in Table 1. The shortest path of the crisp network is $1 \rightarrow 4 \rightarrow 10 \rightarrow 14 \rightarrow 17 \rightarrow 18$ with a total length of 34. Assume that the decision-maker has chosen the values of parameters $\Delta_{ij1}$ and $\Delta_{ij2}$ for each imprecise arc length to represent fuzzy numbers, $\tilde{c}_{ij} = (c_{ij} - \Delta_{ij1}, c_{ij}, c_{ij} + \Delta_{ij2})$, as listed in Table 2.

Here, the estimated arc lengths were obtained according to (4), as follows: $c_{12}^{*} = 5.164$, $c_{14}^{*} = 9.309$, $c_{15}^{*} = 7.285$, $c_{16}^{*} = 7.902$, $c_{16}^{*} = 7.902$, $c_{16}^{*} = 7.902$, $c_{23}^{*} = 11.214$, $c_{24}^{*} = 9.428$, $c_{38}^{*} = 6.447$, $c_{3,10}^{*} = 9.845$, $c_{48}^{*} = 8.690$, $c_{4,10}^{*} = 7.124$, $c_{56}^{*} = 6.933$, $c_{57}^{*} = 12.107$, $c_{59}^{*} = 8.626$, $c_{64}^{*} = 5.130$, $c_{6,60}^{*} = 7.726$, $c_{6,10}^{*} = 8.609$, $c_{6,14}^{*} = 15.047$, $c_{79}^{*} = 7.166$, $c_{7,13}^{*} = 12.005$, $c_{8,11}^{*} = 12.214$, $c_{8,12}^{*} = 6.128$, $c_{9,13}^{*} = 9.128$, $c_{10,11}^{*} = 8.056$, $c_{10,14}^{*} = 2.805$, $c_{10,15}^{*} = 12.064$, $c_{11,15}^{*} = 12.214$, $c_{11,15}^{*} = 6.128$, $c_{9,13}^{*} = 9.128$, $c_{10,11}^{*} = 8.056$, $c_{10,14}^{*} = 2.805$, $c_{10,15}^{*} = 12.064$, $c_{11,15}^{*} =
Table 1. The arc lengths of the network in Figure 7

<table>
<thead>
<tr>
<th>arc</th>
<th>length</th>
<th>arc</th>
<th>length</th>
<th>arc</th>
<th>length</th>
<th>arc</th>
<th>length</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>5</td>
<td>(5,6)</td>
<td>7</td>
<td>(9,12)</td>
<td>6</td>
<td>(14,15)</td>
<td>6</td>
</tr>
<tr>
<td>(1,4)</td>
<td>9</td>
<td>(5,7)</td>
<td>12</td>
<td>(9,13)</td>
<td>9</td>
<td>(14,16)</td>
<td>8</td>
</tr>
<tr>
<td>(1,5)</td>
<td>7</td>
<td>(5,9)</td>
<td>9</td>
<td>(10,11)</td>
<td>8</td>
<td>(14,17)</td>
<td>9</td>
</tr>
<tr>
<td>(1,6)</td>
<td>8</td>
<td>(6,4)</td>
<td>5</td>
<td>(10,14)</td>
<td>3</td>
<td>(15,17)</td>
<td>7</td>
</tr>
<tr>
<td>(2,3)</td>
<td>11</td>
<td>(6,9)</td>
<td>8</td>
<td>(10,15)</td>
<td>12</td>
<td>(15,18)</td>
<td>10</td>
</tr>
<tr>
<td>(2,4)</td>
<td>9</td>
<td>(6,10)</td>
<td>9</td>
<td>(11,15)</td>
<td>7</td>
<td>(16,17)</td>
<td>8</td>
</tr>
<tr>
<td>(3,8)</td>
<td>7</td>
<td>(6,14)</td>
<td>15</td>
<td>(11,18)</td>
<td>13</td>
<td>(17,18)</td>
<td>6</td>
</tr>
<tr>
<td>(3,10)</td>
<td>10</td>
<td>(7,9)</td>
<td>7</td>
<td>(12,14)</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4,8)</td>
<td>9</td>
<td>(7,13)</td>
<td>12</td>
<td>(12,16)</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4,10)</td>
<td>7</td>
<td>(8,11)</td>
<td>13</td>
<td>(13,16)</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7. Another example of the network for finding the shortest path

Table 2. The values of parameters chosen by the decision-maker

<table>
<thead>
<tr>
<th>arc</th>
<th>$\Delta_{ij_1}, \Delta_{ij_2}$</th>
<th>arc</th>
<th>$\Delta_{ij_1}, \Delta_{ij_2}$</th>
<th>arc</th>
<th>$\Delta_{ij_1}, \Delta_{ij_2}$</th>
<th>arc</th>
<th>$\Delta_{ij_1}, \Delta_{ij_2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>{0.2, 0.7}</td>
<td>(5,6)</td>
<td>{1.2, 1.6}</td>
<td>(9,12)</td>
<td>{0.6, 1.2}</td>
<td>(14,15)</td>
<td>{1.5, 1.2}</td>
</tr>
<tr>
<td>(1,4)</td>
<td>{0.5, 1.5}</td>
<td>(5,7)</td>
<td>{0.5, 1.0}</td>
<td>(9,13)</td>
<td>{0.6, 1.2}</td>
<td>(14,16)</td>
<td>{0.3, 1.2}</td>
</tr>
<tr>
<td>(1,5)</td>
<td>{0.2, 1.0}</td>
<td>(5,9)</td>
<td>{0.9, 0.4}</td>
<td>(10,11)</td>
<td>{1.4, 2.2}</td>
<td>(14,17)</td>
<td>{1.4, 1.2}</td>
</tr>
<tr>
<td>(1,6)</td>
<td>{0.3, 0.2}</td>
<td>(6,4)</td>
<td>{0.8, 1.5}</td>
<td>(10,14)</td>
<td>{0.6, 0.4}</td>
<td>(15,17)</td>
<td>{0.5, 0.3}</td>
</tr>
<tr>
<td>(2,3)</td>
<td>{1.0, 2.0}</td>
<td>(6,9)</td>
<td>{0.8, 0.5}</td>
<td>(10,15)</td>
<td>{0.3, 0.6}</td>
<td>(15,18)</td>
<td>{0.6, 1.4}</td>
</tr>
<tr>
<td>(2,4)</td>
<td>{0.3, 1.5}</td>
<td>(6,10)</td>
<td>{1.2, 0.8}</td>
<td>(11,15)</td>
<td>{1.3, 1.0}</td>
<td>(16,17)</td>
<td>{1.3, 1.6}</td>
</tr>
<tr>
<td>(3,8)</td>
<td>{1.2, 0.4}</td>
<td>(6,14)</td>
<td>{0.6, 1.0}</td>
<td>(11,18)</td>
<td>{0.6, 0.3}</td>
<td>(17,18)</td>
<td>{1.3, 1.6}</td>
</tr>
<tr>
<td>(3,10)</td>
<td>{0.6, 0.5}</td>
<td>(7,9)</td>
<td>{0.4, 1.0}</td>
<td>(12,14)</td>
<td>{0.4, 1.2}</td>
<td></td>
<td>{0.2, 1.1}</td>
</tr>
<tr>
<td>(4,8)</td>
<td>{1.2, 1.0}</td>
<td>(7,13)</td>
<td>{0.4, 0.6}</td>
<td>(12,16)</td>
<td>{0.4, 0.8}</td>
<td></td>
<td>{0.2, 1.1}</td>
</tr>
<tr>
<td>(4,10)</td>
<td>{0.2, 0.6}</td>
<td>(8,11)</td>
<td>{2.0, 1.0}</td>
<td>(13,16)</td>
<td>{1.2, 0.8}</td>
<td></td>
<td>{0.2, 1.1}</td>
</tr>
</tbody>
</table>

$6.630, c_{11,18} = 12.764, c_{12,14} = 4.247, c_{12,16} = 10.085, c_{13,16} = 7.609, c_{14,15} = 5.592, c_{14,16} = 8.307, c_{14,17} = 8.652, c_{15,17} = 6.824, c_{15,18} = 10.209, c_{16,17} = 7.873$ and $c_{17,18} = 6.326$. This example focused on observing the effectiveness of the two-population scheme. In the first experiment, the GA program was run 30 times. The average feasible-to-infeasible count calculated over 30 runs was 2129.833. In contrast, the average infeasible-to-feasible count calculated over 30 runs was only 0.8.

The second experiment considered the mutation rate. At each mutation rate, the GA program was run 10 times to obtain 10 infeasible-to-feasible counts, and the average count
was then computed from these 10 counts. Nineteen different mutation rates were used to obtain 19 average counts. The mean of these 19 average counts was 0.5526. Experimental results demonstrate that the mutation rate had no significant influence on the infeasible-to-feasible counts obtained. Conversely, the average feasible-to-infeasible counts increased proportionately to the mutation rate, as shown in Figure 8. In this experiment, 10 feasible-to-infeasible counts were obtained, and used to calculate the average count. Similarly, 19 average feasible-to-infeasible counts were obtained at 19 different mutation rates. The mean of these 19 average feasible-to-infeasible counts was 6540.89. Obviously, the average feasible-to-infeasible counts vastly outnumbered the average infeasible-to-feasible counts.

**Figure 8.** The average feasible-to-infeasible counts yields a linear growth function upon increasing the mutation rate

In the third experiment, the mutation rate was set at a fixed value of 0.003, but the experiment used different crossover rates to evaluate the effectiveness of the two-population scheme. The curve in Figure 9 represents nine average feasible-to-infeasible counts obtained at nine different crossover rates. Each average feasible-to-infeasible count was calculated over 10 runs. The mean of nine average feasible-to-infeasible counts was 635.8. The crossover rate was set to 0.9, the mutation rate was set to 0.003, and the population...
Figure 10 shows the best paths obtained using different numbers of generations starting from 1000 to 9000 with an increment of 1000. The average value of these best paths was 34.476, which is 1.4% longer than the crisp optimal path of 34.

The proposed approach can tackle larger-sized problems because of the characteristic properties of the search process in the shortest path problem when using the two-population scheme. The infeasible population is a continuing source of variation in the feasible population throughout the runs. The feasible population definitely converges as its variance minimizes.

Finally, the performance of the two-population GA compared to that of the one-population scheme is investigated. In these experiments, a normal GA was compared with the same total numbers of generations, split equally among the populations. Thus, for instance, the two-population GA in which each population was run for 3000 generations corresponds to 6000 generations in a normal GA. Therefore, the proposed two-population GA required only a slightly longer computational time than a one-population GA. However, the two-population GA yielded superior solutions to the one-population GA. The average value for the best paths obtained using different numbers of generations starting from 1000 to 9000 with an increment of 1000 in the two-population scheme was 34.476. The average value of the best paths obtained in a normal GA under the same parameter settings was 36.153, which is 4.86% longer than that obtained from a one-population GA.

5. Conclusions. This study investigated the use of a two-stage two-population GA approach to solve the shortest path problem with imprecise arc lengths. In the first stage, the GA program simulates the membership grade of a fuzzy number for representing each imprecise arc length. After calculating the estimated value of each imprecise arc length, the first stage generates a defuzzified instance of the problem. In the second stage, the two-population scheme is used to find the best path from the source to the destination. Experimental results show that the feasible population always converges to a fixed value after a number of generations, while the infeasible population never converges.

The average number of feasible-to-infeasible counts is large, but the average number of infeasible-to-feasible counts is relatively small. Although the infeasible-to-feasible count is small, it helps the GA to find better solutions. The mutation rates have no significant influence on the average infeasible-to-feasible counts obtained. Instead, the resulting average feasible-to-infeasible counts display a direct linear response as the mutation rate increases. Considering the characteristic properties of the search process and the issue of scalability in the shortest path problem, we conclude that a two-population scheme
can obtain better performance and efficiency than a one-population scheme. The feasible population converges as its variance minimizes, while the infeasible population never converges. The infeasible population is a continuing source of variation in the feasible population throughout the runs. Hence, the proposed approach effectively tackles many larger-sized problems. The effectiveness of the proposed GA was analyzed in two different ways: using the same number of generations but with different population sizes, and using the same population size but with different numbers of generations. Experimental results show that the performance may be improved in either way. The percentage differences between the best path found by both methods and the fuzzy approach were obtained. Both methods obtained better results than the fuzzy approach. Future study could consider two important issues. First, the performance of other selection strategies and crossover operators could be compared to the method and mechanism proposed here. Second, the two-population scheme could be improved, and compared systematically with other fuzzy approaches in terms of efficiency.

REFERENCES