Solving the Probabilistic Traveling Salesperson Problem with Profits (pTSPP) with a Noise-aware Evolutionary Multiobjective Optimization Algorithm

Bingchun Zhu
Department of Computer Science
University of Massachusetts Boston
Boston, MA 02125
numchun@cs.umb.edu

Junichi Suzuki
Department of Computer Science
University of Massachusetts Boston
Boston, MA 02125
jxs@cs.umb.edu

Pruet Boonma
Department of Computer Engineering
Chiang Mai University
Chiang Mai, Thailand
pruet@eng.cmu.ac.th

***Abstract*—This paper studies an evolutionary algorithm to solve a new multiobjective optimization problem (MOP), the Probabilistic Traveling Salesperson Problem with Profits (pTSPP), which has noisy objective functions. As a variant of TSP, many real-world noisy MOPs can be reduced to pTSPP. The proposed algorithm leverages a novel noise-aware dominance operator, called the α-dominance operator. The operator statistically estimates the impacts of noise on objective functions and judges which solution candidates are superior/inferior to the others. Unlike existing noise-aware dominance operators, the α-dominance operator assumes no noise distributions. Thus, it is well applicable to various real-world noisy MOPs that follow unknown noise distributions. Experimental results show that the α-dominance operator effectively reveals the dominance relationships among solution candidates, aids to obtain quality solutions to pTSPP and outperforms existing noise-aware dominance operators.**

I. Introduction

This paper focuses on noisy multiobjective optimization problems (MOPs), which are formalized as follows:

$$\left.\begin{array}{rl} \min & F(\vec{x}) = [f_1(\vec{x}) + \epsilon_1, \cdots, f_m(\vec{x}) + \epsilon_m]^T \in \mathcal{O} \\ & \text{subject to } \vec{x} = [x_1, x_2, \cdots, x_n]^T \in \mathcal{S} \end{array}\right\} \quad (1)$$

$\mathcal{S}$ denotes the decision variable space. $\vec{x} \in \mathcal{S}$ denotes a solution candidate that consists of n decision variables. $F : \mathbb{R}^n \rightarrow \mathbb{R}^m$ consists of m real-value objective functions, which produce the objective values of $\vec{x}$ in the objective space $\mathcal{O}$. The goal of MOPs is to find an individual(s) that minimizes objective values with respect to F.

In MOPs, objective functions often conflict with each other. Thus, there exists rarely a single solution that is optimum with respect to all objectives. As a result, MOPs often aim to find the optimal trade-off solutions, or Pareto-optimal solutions, by balancing the trade-offs among conflicting objectives. A notion of *dominance* plays an important role to seek Pareto optimality in MOPs [1]. A solution candidate $\vec{x} \in \mathcal{S}$ is said to *dominate* another solution candidate $\vec{y} \in \mathcal{S}$ (denoted by $\vec{x} \succ \vec{y}$) iif the both of the following conditions are hold.

- $f_i(\vec{x}) \leq f_i(\vec{y}) \ \forall \ i \ = 1, \cdots, m$
- $f_i(\vec{x}) < f_i(\vec{y}) \ \exists \ i \ = \ 1, \cdots, m$

In Equation 1, ϵ_i is a random variable that represents noise in the i-th objective function. $\epsilon_i \neq 0$ in noisy MOPs. This means that each objective function can yield different objective values for the same solution candidate from time to time.

This paper formulates a new noisy MOP, the probabilistic traveling salesperson problem with profits (pTSPP), which can derive a number of real-world optimization problems. pTSPP is a combination of existing two variants of the traveling salesperson problem (TSP): the probabilistic TSP (pTSP) [2] and the TSP with profits (TSPP) [3].

Noise in objective functions often interferes with a dominance operator, which determines dominance relationships among solution candidates. Defects in the operator significantly degrade the performance (e.g., convergence velocity) to solve MOPs [4], [5]. To address this issue, this paper proposes a notion of *noise-aware dominance*, called *α-dominance*, and studies the α-dominance operator for evolutionary multiobjective optimization algorithms (EMOAs). An EMOA uses a population of individuals, each of which represents a solution candidate. It evolves individuals through generations and seeks the Pareto optimal solutions to a MOP. The α-dominance operator takes objective value samples of given two individuals, estimates the impacts of noise on the samples and determines whether it is statistically confident enough to judge a dominance relationship between the two individuals.

This paper describes the design of the α-dominance operator and evaluates it with pTSPP. Experimental results demonstrate that the α-dominance operator reliably performs dominance operation to solve pTSPP and outperforms existing noise-aware dominance operators.

II. Probabilistic Traveling Salesperson Problem with Profits (pTSPP)

pTSPP is defined on a fully-connected graph $G = (V, E)$.

- $V = \{v_0, v_1, v_2, ..., v_n\}$ is the set of vertices in G, where v_0 is the depot. $V' = V - \{v_0\}$ is the set of n non-depot vertices. This paper assumes that vertices are stationary, and $|V|$ does not change dynamically.

- $E = \{v_i, v_j | v_i, v_j \in V; i \neq j\}$ is the set of edges. Each edge $\{v_i, v_j\} \in E$ has an associated cost c_{v_i,v_j}.
- Each vertex $v_i \in V'$ maintains a visiting probability p_{v_i}, which represents the probability that v_i is visited. $p_{v_i} \in [0.0, 1.0]$. The visiting probability of the depot $p_{v_0} = 1.0$.
- Each vertex $v_i \in V'$ has an associated profit $\rho_{v_i} > 0$. The depot's profit $\rho_{v_0} = 0$.
- R is a route, which is a sequence of vertices, starting and ending with v_0. R may not contain all the vertices in V': $|R| \leq |V'| + 2$. No redundant vertices exist in R except v_0. (A non-depot vertex is not visited more than once.) R is an *a posteriori* route. The salesperson uses it to decide, a posteriori, which vertices he/she actually visits based on the visiting probabilities associated with the vertices in R.

pTSPP is to find the Pareto-optimal routes with respect to the following two objectives.

- *Cost*: The total traveling cost that the salesperson incurs by visiting vertices in a route. This objective is to be minimized. It is computed as:

$$f_{cost} = \sum_{v_n, v_{n'} \in R} p_{v_n} p_{v_{n'}} c_{v_n, v_{n'}} \tag{2}$$

  where $v_{n'}$ is the next vertex of v_n in R.
- *Profit*: The total profit that the salesperson gains by visiting vertices in a route. This objective is to be maximized. It is computed as:

$$f_{profit} = \sum_{v_n \in R} p_{v_n} \rho_{v_n} \tag{3}$$

Two objectives in pTSPP conflict with each other. For example, a shorter route (i.e., a route containing a smaller number of vertices) yields a lower cost and a lower profit. On the contrary, a longer route (i.e., a route containing a larger number of vertices) yields a higher cost and a higher profit.

pTSPP considers noise in its objective functions. Following the notations in Equation 1, pTSPP is formulated as follows.

$$\left.\begin{aligned} \min\ F(R) = [f_{cost}(R) + \epsilon_{cost}, \tfrac{1}{f_{profit}(R)} + \epsilon_{profit}]^T \in \mathcal{O} \\ \text{subject to } R = [v_0, \cdots, v_n, v_{n'} \cdots, v_0] \in \mathcal{S} \end{aligned}\right\} \tag{4}$$

ϵ_{cost} and ϵ_{profit} denote noise in the cost and profit objective functions, respectively.

As mentioned in Section I, pTSPP is a combination of pTSP [2] and TSPP [3]. pTSP is to find the minimum-cost route. Each vertex requires a visit of the salesperson with an associated visiting probability. TSPP is to find the optimal route with respect to profit as well as cost. The salesperson can visit a subset of given vertices. pTSPP is similar to pTSP in that both consider cost and visiting probability. It extends pTSP by considering profit as an extra objective and computing the total profit of a route based on the profit and visiting probability associated with each vertex (Equation 3). pTSPP is similar to TSPP in that both consider cost and profit as objectives. However, unlike TSPP, pTSPP considers a visiting probability for each vertex.

Many real-world noisy MOPs can be reduced to pTSPP as various real-world optimization problems can be reduced to pTSP and TSPP [2], [3], [6], [7]. For example, pTSPP can represent noisy MOPs in transportation planning, supply chain networks and data routing/gathering in computer networks.

III. Related Work

The α-dominance operator was proposed first in [8] and evaluated with several test problems such as ZDT1, ZDT2 and ZDT3 [9]. This paper evaluates it with a new noisy MOP, pTSPP, which can derive more realistic optimization problems.

pTSP and TSPP have been studied extensively and used to model many real-world problems in different fields [3]. Early pTSP studies adopted the heuristics for TSP (e.g., nearest neighbor, savings heuristic, k-OPT exchanges and 1-shift) and modified them to solve pTSP (e.g., [10]). Recent studies often focus on meta-heuristics, such as ant colony optimization algorithms [11] and evolutionary algorithms [12], [13], in favor of their global search capabilities. However, these algorithms are not applicable for pTSPP because pTSPP is a multiobjective optimization problem.

TSPP is a multiobjective optimization problem; however, a number of existing work have attempted to solve it by aggregating multiple objectives into a single fitness function as, for example, a weighted sum of objective values or by considering extra objectives as constraints [14], [15]. These algorithms are not designed to seek Pareto-optimal solutions among conflicting objectives. Moreover, it is not always trivial to manually tune weight values in a fitness function that aggregates multiple objective values.

A few existing work have attempted to solve TSPP with multiobjective optimization algorithms (e.g., [16].) These algorithms better address the characteristics of pTSPP; however, they never consider noise in objective functions.

In the area of EMOAs, there exist several existing work to handle uncertainties in objective functions by modifying NSGA-II's classical dominance operator. All of them assume particular noise distributions in advance. For example, [17]–[19] assume normal distributions. [20] assumes uniform distributions. [21], [22] assume Poisson distributions. Given a noise distribution, these existing work collect objective value samples from each individual ([17], [18], [20]–[22]) or cluster individuals ([19]) in order to determine dominance relationships among individuals. If dominance operators assume noise distributions, they are less applicable for real-world noisy MOPs in which noise follows unknown distributions and noise distributions can dynamically change. In contrast, the α-dominance operator assumes no noise distributions in advance. Instead of estimating each individual's objective values, the α-dominance operator estimates the impacts of noise on objective value samples and determines whether it is statistically confident enough to compare individuals.

[23] divides the decision variable space into hyperspheres and runs an EMOA in each hypersphere. EMOAs strive to

move and guide hyperspheres toward the Pareto optima. Each EOMA uses a classical dominance operator, and its noise handling is based on a simple sampling scheme that uses the average of objective value samples as each individual's objective value. In contrast, the α-dominance operator extends the classical dominance operator with a statistical scheme in order to perform noise handling in a more reliable manner.

[24] considers noisy objective functions in an indicator-based EMOA, called IBEA [25], which exploits performance indicators such as ϵ indicator and hypervolume indicator to rank individuals without using classical dominance relationships. Its noise handling operator takes a set of indicator value samples of each individual and uses their average (and other measures) as the individual's indicator value. Unlike [24], the α-dominance operator examines noise-aware dominance relationships that leverage statistical characteristics of objective value samples.

IV. The Proposed Evolutionary Multiobjective Optimization Algorithm for pTSPP

This section describes the proposed noise-aware EMOA.

A. Individual Representation

In the proposed EMOA, each individual represents a solution candidate: a posteriori route R that contains a sequence of vertices. (See Section II.) Every individual has the depot (v_0) as its first and last element. Figure 1 shows an example individual. Given this route, the salesperson starts with v_0, visits v_3 and its subsequent 7 vertices, and returns back to v_0. Different individuals have different lengths (i.e., different numbers of vertices).

0	3	5	7	2	10	4	9	1	0

Fig. 1. An Example Individual

B. Algorithmic Structure

Algorithm 1 shows the algorithmic structure of evolutionary optimization in the proposed EMOA. It follows the optimization process in NSGA-II, a well-known existing EMOA [26].

At the *0*-th generation, N individuals are randomly generated as the initial population $\mathcal{P}_0$ (Line 2). Each of them contains randomly-selected vertices in a random order. At each generation (g), two parent individuals (p_1 and p_2) are selected from the current population $\mathcal{P}_g$ with binary tournaments (Lines 6 and 7). A binary tournament randomly takes two individuals from $\mathcal{P}_g$, compares them based on α-dominance relationship, and chooses a superior one as a parent.

With the crossover rate P_c, two parents reproduce two offspring with a crossover operator (Lines 8 to 10). Each offspring performs mutation with the mutation rate P_m (Lines 11 to 16). The binary tournament, crossover and mutation operators are executed repeatedly on $\mathcal{P}_g$ to reproduce N offspring. The offspring ($\mathcal{O}_g$) are combined with the parent population $\mathcal{P}_g$ to form $\mathcal{R}_g$ (Line 19).

The environmental selection process follows the reproduction process. N individuals are selected from $2N$ individuals in $\mathcal{R}_g$ as the next generation's population ($\mathcal{P}_{g+1}$). First, the individuals in $\mathcal{R}_g$ are ranked based on their α-dominance relationships. Non-dominated individuals are on the first rank. The i-th rank consists of the individuals dominated only by the individuals on the $(i-1)$-th rank. Ranked individuals are stored in $\mathcal{F}$ (Line 20). $\mathcal{F}_i$ contains the i-th rank individuals.

Then, the individuals in $\mathcal{F}$ move to $\mathcal{P}_{g+1}$ on a rank by rank basis, starting with $\mathcal{F}_1$ (Lines 23 to 26). If the number of individuals in $\mathcal{P}_{g+1} \cup \mathcal{F}_i$ is less than N, $\mathcal{F}_i$ moves to $\mathcal{P}_{g+1}$. Otherwise, a subset of $\mathcal{F}_i$ moves to $\mathcal{P}_{g+1}$. The subset is selected based on the crowding distance metric, which measures the distribution (or diversity) of individuals in the objective space [26] (Lines 27 and 28). The metric computes the distance between two closest neighbors of an individual in each objective and sums up the distances associated with all objectives. A higher crowding distance means that an individual in question is more distant from its neighboring individuals in the objective space. In Line 27, the individuals in $\mathcal{F}_i$ are sorted from the one with the highest crowding distance to the one with the lowest crowding distance. The individuals with higher crowding distance measures have higher chances to be selected to $\mathcal{P}_{g+1}$ (Line 28).

Algorithm 1 Optimization Process in the Proposed EMOA

```
1:  g = 0;
2:  P_g = Randomly generated N individuals;
3:  while g < MAX-GENERATION do
4:      O_g = ∅;
5:      while |O_g| < N do
6:          p_1 = tournament(P_g)
7:          p_2 = tournament(P_g)
8:          if random() ≤ P_c then
9:              {o_1, o_2} = crossover(p_1, p_2)
10:         end if
11:         if (random() ≤ P_m) then
12:             o_1 = mutation(o_1)
13:         end if
14:         if random() ≤ P_m then
15:             o_2 = mutation(o_2)
16:         end if
17:         O_g = {o_1, o_2} ∪ O_g
18:     end while
19:     R_g = P_g ∪ O_g
20:     F = sortByDominationRanking(R_g)
21:     P_{g+1} = {∅}
22:     i = 1
23:     while |P_{g+1}| + |F_i| ≤ N do
24:         P_{g+1} = P_{g+1} ∪ F_i
25:         i = i + 1
26:     end while
27:     sortByCrowdingDistance(F_i)
28:     P_{g+1} = P_{g+1} ∪ F_i[1 : (N − |P_{g+1}|)]
29:     g = g + 1
30: end while
```

C. Crossover

The proposed EMOA adopts partially-mapped crossover (PMX) as its crossover operator. PMX was originally proposed

to solve TSP [27]. It is known that PMX effectively works for TSP and its variants [27], [28].

PMX first selects two crossover points on parent individuals at random. A sub-route surrounded by the two crossover points is called a mapping section. In an example in Figure 2, parent 1's mapping section is [2, 8, 7, 3], and parent 2's is [3, 9, 4, 13]. Given two mapping sections, mapping relationships are formed by paring elements in the mapping sections on a position by position basis. In Figure 2, the first elements in two mapping sections, 3 and 2, are paired; 3–2 is the first mapping relationship. Similarly, three other mapping relationships, 9–8, 4–7 and 13–3, are formed. In order to reproduce two offspring from two parents, mapping sections are exchanged between parents. In Figure 2, parent 1's mapping section is replaced with parent 2's. (Note that Figure 2 does not show the other proto-offspring.) If proto-offspring has duplicate vertices across its mapping section and the other section, PMX replaces each duplicate vertex with its counterpart described in a mapping relationship. In Figure 2, 4, 9 and 13 are duplicated. Given a mapping relationship of 7–4, 4 is replaced with 7 in the non-mapping section. (Replacements always occur in the non-mapping section.) 9 is replaced with 8 with a mapping relationship of 9–8. 13 is replaced with 2 by referencing two mapping relationships (13–3 and 3–2) recursively.

The length of each individual is variable as described in Section IV-A. When two parents have different lengths, their mapping section is constrained not to exceed a shorter parent's length. In Figure 2, where parent 2 is shorter, the mapping section's length is always shorter than its length.

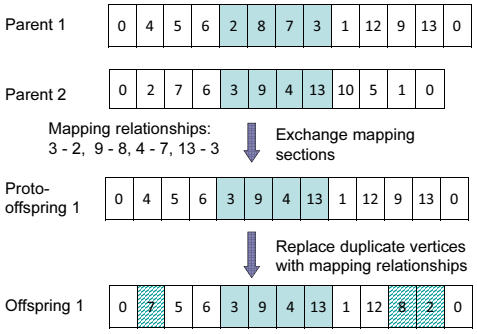


Fig. 2. An Example Crossover (PMX) Process

D. Mutation

The proposed EMOA provides a multi-mode mutation operator to alter reproduced offspring. The operator has four modes and selects one of them at a time randomly.

1) *Add*: randomly chooses a vertex from unselected vertices and inserts it to a randomly-selected position in a route (Figure 3(a)).
2) *Delete*: removes a randomly-selected vertex from a route (Figure 3(b)).
3) *Exchange*: randomly chooses a vertex in a route and replaces it with one of unselected vertices (Figure 3(c)). The unselected vertex is also chosen at random.
4) *Swap*: exchanges the positions of two randomly-selected verticies in a route (Figure 3(d)).

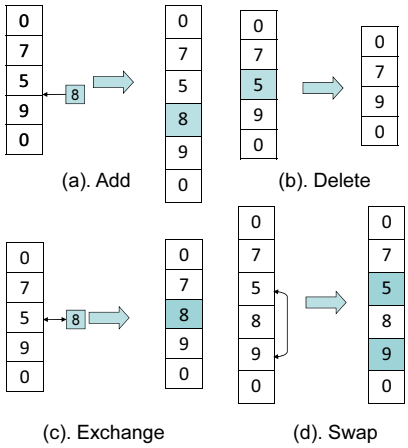


Fig. 3. Example Mutation Modes

E. α-Dominance

This section describes the notion of α-dominance and the design of the α-dominance operator. α-dominance is a new dominance relationship that extends a classical dominance relationship described in Section I. It takes objective value samples of given two individuals, estimates the impacts of noise on the samples, and determines whether it is statistically confident enough to judge which one is superior/inferior between the two individuals. The α-dominance operator is used in the parent selection operator (Lines 6 and 7 in Algorithm 1) and individual ranking operator (Line 20 in Algorithm 1).

With the notion of α-dominance, individual A is said to α-*dominate* individual B (denoted by $A \succ_\alpha B$), iif:

- A's and B's objective value samples are classifiable with a statistical confidence level of α, and
- $\mathcal{C}(A,B) = 1 \wedge \mathcal{C}(B,A) < 1$.

In order to examine the first condition, the α-dominance operator classifies A's and B's objective value samples with Support Vector Machine (SVM), and measures a classification error. The error (e) is computed as the ratio of the number of missclassified samples to the total number of samples. Then, the α-dominance operator computes the classification error's confidence interval (e_{int}):

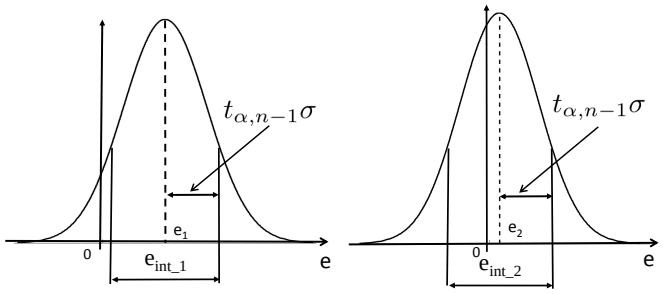
$$e_{int} = e \pm t_{\alpha,n-1}\sigma \tag{5}$$

$t_{\alpha,n-1}$ denotes a single-tail t-distribution with α confidence level and $n-1$ degrees of freedom. n denotes the total number of samples. σ is the standard deviation of e. It is approximated as follows.

$$\sigma \cong \sqrt{\frac{e}{n}} \tag{6}$$

If e_{int} does not span zero, the α-dominance operator judges that a classification error is significant and A's and B's samples are not classifiable with the confidence level of α.

This means that the aforementioned first condition is not hold. As a result, the α-dominance operator determines that A and B do not α-dominate each other. Figure 4(a) shows an example classification error (e_1) and its confidence interval (e_{int_1}). Since the confidence interval does not span zero (i.e., $e_1 - \frac{e_{int_1}}{2} > 0$), e_1 is judged to be significant.



(a). Significant Classification Error (b). Insignificant Classification Error

Fig. 4. Classification Error's Distributions

If e_{int} spans zero, the α-dominance operator judges that a classification error is not significant and A's and B's samples are classifiable with the confidence level of α. This means that the aforementioned first condition is hold. Thus, the α-dominance operator examines the second condition. In an example shown in Figure 4(b), e_2 is not judged to be significant because e_{int_2} spans zero (i.e., $e_2 - \frac{e_{int_2}}{2} < 0$).

In order to examine the second condition, the α-dominance operator measures $\mathcal{C}$-metric [29] with a classical notion of dominance ($\succ$) described in Section I. $\mathcal{C}(A, B)$ denotes the fraction of individual B's samples that at least one sample of individual A dominates:

$$\mathcal{C}(A, B) = \frac{|\{b \in B \mid \exists a \in A : a \succ b\}|}{|B|} \tag{7}$$

If $\mathcal{C}(A, B) = 1$, all of B's samples are dominated by at least one sample of A. If $\mathcal{C}(B, A) < 1$, not all of A's samples are dominated by at least one sample of B. The α-dominance operator determines $A \succ_\alpha B$ if $\mathcal{C}(A, B) = 1$ and $\mathcal{C}(B, A) < 1$. If $\mathcal{C}(A, B) < 1$ and $\mathcal{C}(B, A) < 1$, the operator determines that A and B are non-α-dominated.

Figures 5 and 6 show three examples to determine the α-dominance relationship between two individuals, A and B, with respect to two objectives, f_1 and f_2, to be minimized. Individual A and B have seven samples each. In each example, SVM classifies these 14 samples with a classification vector.

In Figure 5, five samples (two for A and three for B) are missclassified; $e = \frac{5}{14}$ (0.357). Thus, $\sigma \cong \sqrt{\frac{0.357}{14}} = 0.160$. Assuming the confidence level α of 95%, $e_{int} = 0.357 \pm 1.771 * 0.160 = 0.357 \pm 0.283$. Since e_{int} does span zero (i.e., $0.357 - 0.283 > 0$), A's and B's samples are not classifiable with the confidence level of 95%. The aforementioned first condition is not hold. Therefore, the α-dominance operator determines A and B are non-α-dominated in Figure 5.

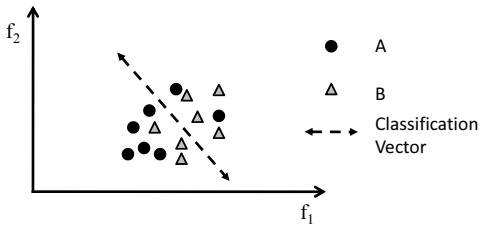


Fig. 5. An Example Classification of Two Individuals (A and B)

In both Figures 6(a) and 6(b), two samples are missclassified. $e = \frac{2}{14}$ (0.143) and $\sigma \cong \sqrt{\frac{0.143}{14}} = 0.1$. Under $\alpha = 0.95$, $e_{int} = 0.143 \pm 1.771 * 0.1 = 0.143 \pm 0.1771$. e_{int} spans zero (i.e., $0.143 - 0.1771 < 0$); therefore, A's and B's samples are classifiable with the confidence level of 95%. This means that the aforementioned first condition is hold in both Figures 6(a) and 6(b).

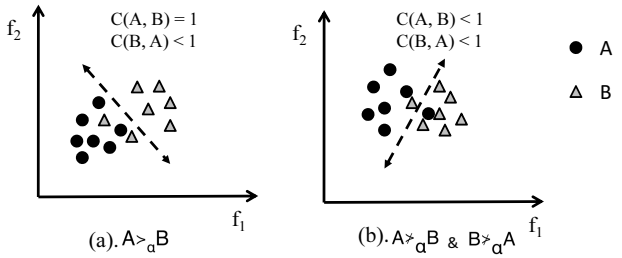


Fig. 6. An Example Process to determine the α-Dominance Relationship between two individuals (A and B)

In Figure 6(a), $\mathcal{C}(A, B) = 1$ and $\mathcal{C}(B, A) = 1/7 < 1$. This means that the aforementioned second condition is hold. As a result, the α-dominance operator concludes $A \succ_\alpha B$ in Figure 6(a). In Figure 6(b), $\mathcal{C}(A, B) = 6/7 < 1$ and $\mathcal{C}(B, A) = 1/7 < 1$. The second condition is not hold. Therefore, the α-dominance operator concludes that A and B are non-α-dominated in Figure 6(b).

Algorithm 2 shows pseudo code of the α-dominance operator. $\mathcal{A}$ and $\mathcal{B}$ denote individual A's and B's samples, respectively. $\mathcal{A}'$ and $\mathcal{B}'$ denote two clusters of samples classified by SVM.

V. Experimental Evaluation

This section evaluates the proposed EMOA, particularly its α-dominance operator, through experiments with a test problem that is built based on a TSP instance called pr226. This TSP instance is obtained from TSPLIB[1] [30]. pr226 contains 226 vertices in a graph. It is customized in this evaluation study so that each vertex maintains a profit and a visiting probability. The value ranges of a profit and a visiting probability are [1.0, 100.0] and [0.0,1.0], respectively. Both values are assigned to each vertex at a uniformly random.

Noise is generated and injected to each of two objective functions every time it is evaluated, as shown in Equation 4.

[1]http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

Algorithm 2 Pseudocode of the α-Dominance Operator

function alphaDominance($\mathcal{A}$, $\mathcal{B}$, α)
$\mathcal{A}'$, $\mathcal{B}'$ = classifyWithSVM($\mathcal{A}$, $\mathcal{B}$)
$count = 0$ // *# of missclassified samples*
for each $x \in \mathcal{A}'$ **do**
    if $x \notin \mathcal{A}$ **then**
        $count = count + 1$
    end if
end for
for each $x \in \mathcal{B}'$ **do**
    if $x \notin \mathcal{B}$ **then**
        $count = count + 1$
    end if
end for
$e = \frac{count}{|\mathcal{A}|+|\mathcal{B}|}$ // *classification error*
$t =$ t-distribution$(\alpha, |\mathcal{A}| + |\mathcal{B}| - 1)$
$e_{int} = e - t \times \sqrt{\frac{e}{|\mathcal{A}|+|\mathcal{B}|}}$
if $e_{int} > 0$ **then**
    // *e is significant.*
    return 0 // *A and B are non-α-dominated.*
else
    // *e is not significant.*
    if $\mathcal{C}(\mathcal{A}, \mathcal{B}) = 1$ **then**
        return 1 // *A α-dominates B.*
    else if $\mathcal{C}(\mathcal{B}, \mathcal{A}) = 1$ **then**
        return -1 // *B α-dominates A.*
    else
        return 0 // *A and B are non-α-dominated.*
    end if
end if
end function

Two types of noise are generated: random noise, which follows continuous uniform distributions, and Gaussian noise, which follow normal distributions. Each noise type has three levels of noise: low, medium and high. Table I illustrates noise configurations. For random noise, each cell of the table shows a pair of the lower and upper bounds of noise values. For Gaussian noise, each cell of the table shows a pair of the mean and variance of noise values.

		Random noise (Uniform distribution)	Gaussian noise (Normal distribution)
Cost	Low	[-320,320]	(0,740)
	Medium	[-1280,1280]	(0,1600)
	High	[-2240,2240]	(0,2560)
Profit	Low	[-2,2]	(0,4)
	Medium	[-8,8]	(0,10)
	High	[-14,14]	(0,16)

TABLE I
NOISE CONFIGURATIONS

The proposed EMOA is configured with a set of parameters shown in Table II. It is called NSGA-II-A, or simply A, in this evaluation study because it extends NSGA-II with the α-dominance operator and other operators. In order to evaluate the α-dominance operator, NSGA-II-A is compared with the following three variants of NSGA-II:

- NSGA-II (or simply R): NSGA-II with its crossover and mutation operators replaced by PMX and a mutation operator described in Section IV-D. Its dominance operator takes a set of object value samples of each individual and use their average as the individual's objective value.
- NSGA-II-N (or simply N): NSGA-II with its dominance operator replaced by a noise-aware dominance operator [18]. The operator takes a set of objective value samples of each individual and estimates its objective value by assuming noise follow normal distributions. This algorithm performs the PMX operator and a mutation operator described in Section IV-D.
- NSGA-II-U (or simply U): NSGA-II with its dominance operator replaced by a noise-aware dominance operator [20]. The operator takes a set of objective value samples of each individual and estimates its objective value by assuming noise follow uniform distributions. This algorithm performs the PMX operator and a mutation operator described in Section IV-D.

All algorithms take the same number of samples in each generation and carry out the same number of objective function evaluations in each experiment. All experiments were conducted with jMetal [31]. Every experimental result is obtained and shown based on 20 independent experiments.

Parameter	Value	SVM Parameter	Value
Population size	100	SVM Type	C-support vector classification
Max Generations	500		
Crossover rate	0.9	Kernel	Linear
Mutation rate	0.2	C parameter	1
Samples size	30	Termination criteria	$1e^{-3}$

TABLE II
PARAMETER CONFIGURATIONS

A. Cost and Profit Analysis

Figures 7, 8, 9 and 10 show the objective values that each algorithm's individuals yield at the last (the 500th) generation. Figures 7 and 8 illustrate the cost and profit distributions under Gaussian noise. Figure 7 depicts that, under Gaussian noise, NSGA-II-A is the best among four algorithms in the median and lowest cost metrics under all noise levels. It is also the best in the median and highest profit metrics under low and high noise levels (Figure 8). NSGA-II-N does not perform well except under the mid-level noise, although its dominance operator expects Gaussian noise in advance. NSGA-II-R yields the worst results in most cases due to its simple sampling scheme. The cost distributions are comparable among four algorithms.

Figures 9 and 10 show the cost and profit distributions under random noise. Their results are qualitatively similar to the ones in Figures 7 and 8. NSGA-II-A yields the best median and lowest cost under low noise level (Figure 9). It also yields the lowest cost under the mid-level noise. In Figure 10, NSGA-II-A yields the best median and highest profit under medium noise level. It also yields the best median profit under high-level noise. In the other cases, NSGA-II-A is comparable with NSGA-II-U, which expects random noise in advance.

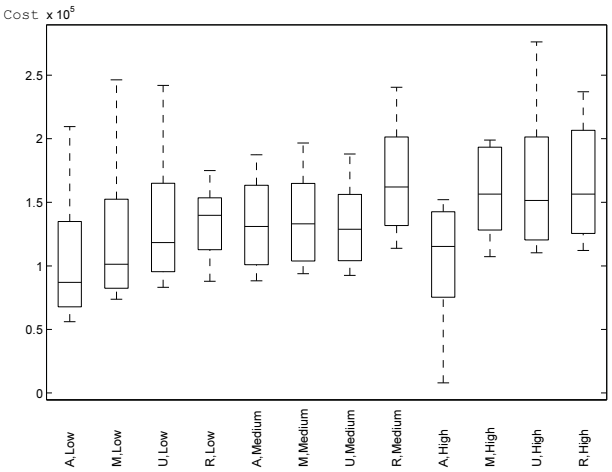


Fig. 7. Cost Distributions under Gaussian Noise

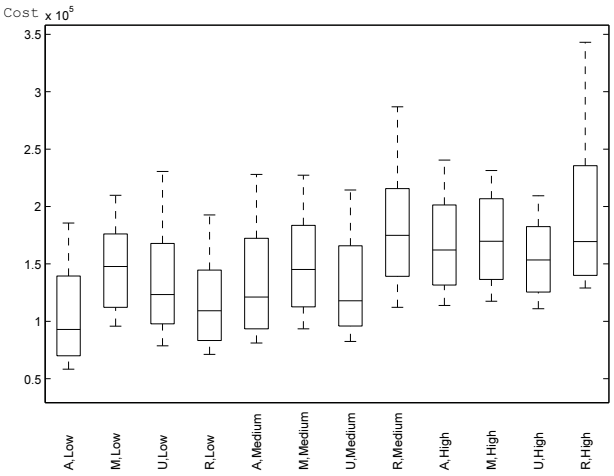


Fig. 9. Cost Distributions under Random Noise

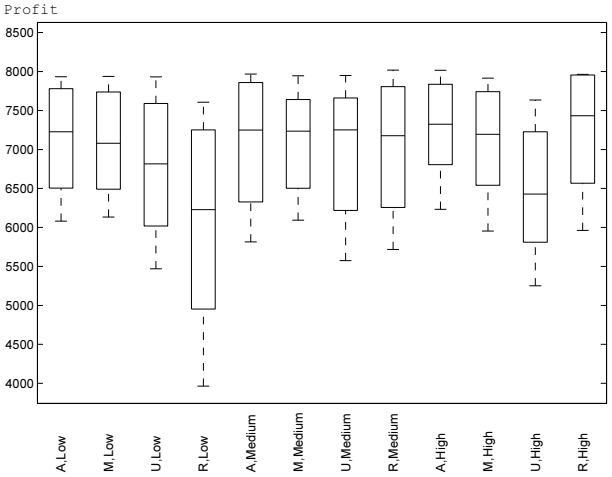


Fig. 8. Profit Distributions under Gaussian Noise

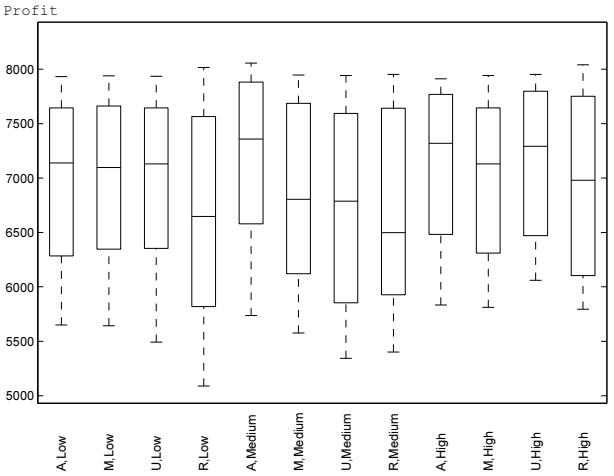


Fig. 10. Profit Distributions under Random Noise

The distributions of individuals are comparable among four algorithms.

B. Algorithm Comparison with C-metric

Tables III and IV show the $\mathcal{C}$-metric measures [29] to compare NSGA-II-A with the other three algorithms under Gaussian and random noise, respectively. This experiment uses 30 objective value samples for each individual at the last generation (3,000 samples in total).

Under Gaussian noise (Tables III), $\mathcal{C}$(NSGA-II-A, NSGA-II) $>$ $\mathcal{C}$(NSGA-II, NSGA-II-A) in all three noise levels. (A bold font face is used to indicate a higher $\mathcal{C}$-metric value between $\mathcal{C}$(NSGA-II-A, NSGA-II) and $\mathcal{C}$(NSGA-II, NSGA-II-A).) This means that NSGA-II-A outperforms NSGA-II in all three noise levels. Under mid-level noise, NSGA-II produces no individuals that dominate the ones produced by NSGA-II-A. Under high-level noise, 40.1% of NSGA-II's individual samples (1,203 samples) are dominated by at least one sample of NSGA-II-A. In comparison between NSGA-II-A and NSGA-II-U, $\mathcal{C}$(NSGA-II-A, NSGA-II-U) $>$ $\mathcal{C}$(NSGA-II-U, NSGA-II-A) in low and high noise levels. This means that NSGA-II-A outperforms NSGA-II-U under low-level and high-level noise and the two algorithms tie under mid-level noise. In comparison between NSGA-II-A and NSGA-II-N, NSGA-II-A outperforms NSGA-II-N in high noise level and the two algorithms tie in the other two noise levels. Even though NSGA-II-N is designed to handle Gaussian noise, it fails to produce individuals that dominate NSGA-II-A's in all noise levels.

Under random noise (Tables IV), NSGA-II-A outperforms NSGA-II in all three noise levels. Under high-level noise, 50.8% of NSGA-II's individual samples (1,524 samples) are dominated by at least one sample of NSGA-II-A. In comparison between NSGA-II-A and NSGA-II-U, which is designed to handle random noise, NSGA-II-A outperforms NSGA-II-U under mid-level and high-level noise. The two algorithms tie in low noise level. Under mid-level and high-level noise, NSGA-II-U fails to produce individuals that dominate NSGA-

	Low	Medium	High
$\mathcal{C}$(NSGA-II-A, NSGA-II-R)	$\mathbf{0.405}$	$\mathbf{0.40}$	$\mathbf{0.401}$
$\mathcal{C}$(NSGA-II-R, NSGA-II-A)	7.29e-3	0.00e+00	3.12e-03
$\mathcal{C}$(NSGA-II-A, NSGA-II-U)	$\mathbf{5.71e-03}$	0.00e+00	$\mathbf{3.18e-3}$
$\mathcal{C}$(NSGA-II-U, NSGA-II-A)	2.32e-03	0.00e+00	0.00e+00
$\mathcal{C}$(NSGA-II-A, NSGA-II-N)	0.00e+00	0.00e+00	$\mathbf{5.29e-3}$
$\mathcal{C}$(NSGA-II-N, NSGA-II-A)	0.00e+00	0.00e+00	0.00e+00

TABLE III
$\mathcal{C}$-METRIC MEASURES UNDER GAUSSIAN NOISE

	Low	Medium	High
$\mathcal{C}$(NSGA-II-A, NSGA-II-R)	$\mathbf{0.502}$	$\mathbf{0.510}$	$\mathbf{0.508}$
$\mathcal{C}$(NSGA-II-R, NSGA-II-A)	4.03e-3	4.21e-3	6.11e-03
$\mathcal{C}$(NSGA-II-A, NSGA-II-U)	0.00e+00	$\mathbf{4.38e-03}$	$\mathbf{6.71e-03}$
$\mathcal{C}$(NSGA-II-U, NSGA-II-A)	0.00e+00	0.00e+00	2.25e-3
$\mathcal{C}$(NSGA-II-A, NSGA-II-N)	0.00e+00	$\mathbf{6.77e-03}$	$\mathbf{7.51e-03}$
$\mathcal{C}$(NSGA-II-N, NSGA-II-A)	0.00e+00	2.10e-03	0.00e+00

TABLE IV
$\mathcal{C}$-METRIC MEASURES UNDER RANDOM NOISE

II-A's. In comparison between NSGA-II-A and NSGA-II-N, NSGA-II-A outperforms NSGA-II-N under mid-level and high-level noise. Under low-level and high-level noise, NSGA-II-N produces no individuals that dominate NSGA-II-A's.

Tables III and IV demonstrate that, although the α-dominance operator assumes no noise distributions in advance, NSGAII-A performs well under both Gaussian and random noise. NSGAII-A exhibits higher superiority against the other three algorithms in higher noise levels.

VI. CONCLUSIONS

This paper formulates a noisy multiobjective optimization problem, the Probabilistic Traveling Salesperson Problem with Profits (pTSPP), which contains noise in its objective functions. In order to solve pTSPP, this paper proposes an evolutionary multiobjective optimization algorithm (EMOA) that leverages a novel noise-aware dominance operator, called the α-dominance operator. Experimental results demonstrate that the operator allows the proposed EMOA to effectively obtain quality solutions and it outperforms existing noise-aware dominance operators.

As future work, extended experiments are planned with, for example, extra noise configurations and evaluation metrics. In addition, a noise-aware diversity preservation operator will be studied and integrated with the α-dominance operator.

REFERENCES

[1] N. Srinivas and K. Deb, "Multiobjective function optimization using nondominated sorting genetic algorithms," *Evol. Computat.*, vol. 2, no. 3, 1995.

[2] P. Jaillet, "Probabilistic traveling salesman problem," Ph.D. dissertation, Massachusetts Institution of Technology, 1985.

[3] D. Feillet, P. Dejax, and M. Gendreau, "Traveling salesman problems with profits," *Transportation Science*, vol. 39, no. 2, 2005.

[4] H.-G. Beyer, "Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2–4, 2000.

[5] L. Bianchi, M. Dorigo, L. Gambardella, and W. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," *Natural Computing*, vol. 8, no. 2, pp. 239–287, 2009.

[6] D. Bertsimas and L. H. Howell, "Further results on the probabilistic traveling salesman problem," *European Journal of Operational Research*, vol. 65, no. 1, pp. 68–95, February 1993.

[7] N. Jozefowiez, F. Glover, and M. Laguna, "Multi-objective metaheuristics for the traveling salesman problem with profits," *Journal of Mathematical Modelling and Algorithms*, vol. 7, no. 2, 2008.

[8] P. Boonma and J. Suzuki, "A confidence-based dominance operator in evolutionary algorithms for noisy multiobjective optimization problems," in *Proc. of IEEE Int'l Conf. on Tools with Artificial Intelligence*, 2009.

[9] E. Zitzler, K. Deb, and L. Thieler, "Comparison of multiobjective evolutionary algorithms: Empirical results," *IEEE Trans. Evol. Computat.*, vol. 8, no. 2, 2000.

[10] M. Birattari, P. Balaprakash, T. Sttzle, and M. Dorigo, "Estimation-based local search for the probabilistic traveling salesman problem," in *Proc. of Metaheuristics International Conference*, 2007.

[11] J. Branke and M. Guntsch, "Solving the probabilistic TSP with ant colony optimization," *J. Math. Model. Algorithms*, vol. 3, no. 4, 2004.

[12] Y.-H. Liu, R.-C. Jou, C.-C. Wang, and C.-S. Chiu, "An evolutionary algorithm with diversified crossover operator for the heterogeneous probabilistic TSP," in *Proc. of Int'l Conference on Modeling Decisions for Artificial Intelligence*, 2007.

[13] Y.-H. Liu, "Solving the probabilistic traveling salesman problem based on genetic algorithm with queen selection scheme," in *Traveling Salesman Problem*, F. Greco, Ed. InTech, 2008.

[14] B. Awerbuch, Y. Azar, A. Blum, and S. Vempala, "New approximation guarantees for minimum-weight k-trees and prize-collecting salesmen," *SIAM J. Comput.*, vol. 28, no. 1, 1999.

[15] G. Laporte and S. Martello, "The selective travelling salesman problem," *Discrete Appl. Math.*, vol. 26, no. 2-3, pp. 193–207, 1990.

[16] N. Jozefowiez, F. Glover, and M. Laguna, "Multi-objective metaheuristics for the traveling salesman problem with profits," *Journal of Mathematical Modelling and Algorithms*, vol. 7, pp. 177–195, 2008.

[17] C. K. Goh and K. C. Tan, "Noise handling in evolutionary multi-objective optimization," in *Proc. of IEEE Congress on Evolutionary Computation*, 2006.

[18] H. Eskandari, C. D. Geiger, and R. Bird, "Handling uncertainty in evolutionary multiobjective optimization: SPGA," in *Proc. of IEEE Congress on Evolutionary Computation*, 2007.

[19] M. Babbar, A. Lakshmikantha, and D. E. Goldberg, "A modified NSGA-II to solve noisy multiobjective problems," in *Proc. of ACM Genetic and Evolutinoary Computation Conference*, 2003.

[20] J. Teich, "Pareto-front exploration with uncertain objectives," in *Proc. of Int'l Conf. on Evol. Multi-Criterion Optimization*, 2001.

[21] M. Wormington, C. Panaccione, K. M. Matney, and D. K. Bowen, "Characterization of structures from x-ray scattering data using genetic algorithms," *Phil. Trans. R. Soc. Lond. A*, vol. 357, no. 1761, 1999.

[22] K. Delibrasis, P. Undrill, and G. Cameron, "Genetic algorithm implementation of stack filter design for image restoration," in *Proc. of Vision, Image and Signal Processing*, 1996.

[23] L. T. Bui, H. A. Abbass, and D. Essam, "Localization for solving noisy multi-objective optimization problems," *Evol. Comput.*, vol. 17(3), 2009.

[24] M. Basseur and E. Zitzler, "Handling uncertainty in indicator-based multiobjective optimization," *International Journal of Computational Intelligence Research*, vol. 2, no. 3, 2006.

[25] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Proc. of Int'l Conf. on PPSN*, 2004.

[26] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Proc. of Int'l Conference on PPSN*, 2000.

[27] D. E. Goldbert and R. Lingle, "Alleles, loci, and the traveling salesman problem," in *Proc. of Int'l Conf. on Genetic Algorithms and Their Applications*, 1985.

[28] T. Kellegőz, B. Toklu, and J. Wilson, "Comparing efficiencies of genetic crossover operators for one machine total weighted tardiness problem," *Applied Mathematics and Computation*, vol. 199, no. 2, 2008.

[29] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, 1999.

[30] G. Reinelt, "Tsplib–a traveling salesman problem library," *ORSA Journal on Computing*, vol. 3, no. 4, 1991.

[31] J. Durillo, A. Nebro, and E. Alba, "The jMetal framework for multi-objective optimization: Design and architecture," in *Proc. of IEEE Congress on Evolutionary Computation*, 2010.