Boosting Indicator-based Selection Operators for Evolutionary Multiobjective Optimization Algorithms

Dung H. Phan
Department of Computer Science
University of Massachusetts, Boston
Boston, MA 02125
phdung@cs.umb.edu

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

***Abstract*—Various evolutionary multiobjective optimization algorithms (EMOAs) have adopted indicator-based selection operators that augment or replace dominance ranking with quality indicators. A quality indicator measures the goodness of each solution candidate. Many quality indicators have been proposed with the intention to capture different preferences in optimization. Therefore, indicator-based selection operators tend to have biased selection pressures that evolve solution candidates toward particular regions in the objective space. An open question is whether a set of existing indicator-based selection operators can create a single operator that outperforms those existing ones. To address this question, this paper studies a method to aggregate (or boost) existing indicator-based selection operators. Experimental results show that a boosted selection operator outperforms exiting ones in optimality, diversity and convergence velocity. It also exhibits robustness against different characteristics in different optimization problems and yields stable performance to solve them.**

***Keywords*-Evolutionary multiobjective optimization algorithms, Quality indicators, Boosting**

I. Introduction

This paper studies a new selection operator for evolutionary algorithms to solve multiobjective optimization problems (MOPs). In general, an MOP is described as follows.

$$\left.\begin{array}{r}\text{minimize } F(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \cdots, f_n(\vec{x})]^T \in \mathcal{O} \\ \text{subject to } \vec{x} = [x_1, x_2, \cdots, x_m]^T \in \mathcal{S}\end{array}\right\} \quad (1)$$

$\mathcal{S}$ denotes the decision variable space. $\vec{x} \in \mathcal{S}$ denotes a solution candidate that consists of m decision variables. It is called an *individual* in evolutionary multiobjective optimization algorithms (EMOAs). F consists of n real-value objective functions, which produce the objective values of $\vec{x}$ in the objective space $\mathcal{O}$. The goal of an EMOA is to find an individual(s) that minimize(s) objective values.

In MOPs, there rarely exists a single solution that is optimum with respect to all objectives because objective functions conflict with each other. Thus, EMOAs seek the optimal trade-off individuals, or *Pareto-optimal* individuals, by considering the trade-offs among conflicting objectives. The notion of *dominance* plays an important role to seek Pareto optimality [1]. An individual $\vec{x} \in \mathcal{S}$ is said to *dominate* another individual $\vec{y} \in \mathcal{S}$ iif $f_i(\vec{x}) \leq f_i(\vec{y})\ \forall i = 1, \cdots, n$ and $f_i(\vec{x}) < f_i(\vec{y})\ \exists i = 1, \cdots, n$. EMOAs often rank individuals based on the dominance relationships among them and exploit their ranks in selection operators [1]. This process is called *dominance ranking*.

A research trend in the design space of EMOAs is to adopt *indicator-based selection* operators that augment or replace dominance ranking with quality indicators [2]. A quality indicator measures the goodness of each individual. Recent studies (e.g., [3]) show that indicator-based EMOAs outperform traditional EMOAs that use dominance ranking.

Many quality indicators have been proposed with the intention to capture different preferences in optimization [4]–[7]. Therefore, indicator-based selection operators tend to have biased selection pressures that evolve individuals toward particular regions in the objective space. For example, the hypervolume indicator favors *balanced* individuals that equally balance the trade-offs among all objectives, while the weighted hypervolume indicator favors *extreme* individuals that yield superior performance only in a limited number of objectives [4]. An open question in this context is whether a set of existing indicator-based selection operators can create a single operator that outperforms those existing ones.

In order to address this question, this paper proposes and evaluates a method to aggregate (or boost) existing indicator-based selection operators[1]. This boosting process is carried out with a training problem in which Pareto-optimal solutions are known. Experimental results show that a boosted selection operator outperforms exiting ones in optimality, diversity and convergence velocity. The proposed boosting process can work with a simple training problem, and the boosted operator can effectively solve harder problems. The boosted operator also exhibits robustness against different characteristics in different problems and yields stable performance to solve them.

[1]In this paper, a selection operator means a parent selection operator, which chooses individuals from the population to reproduce offspring.

II. Related Work

To the best of the authors' knowledge, this work is the first attempt to boost selection operators in evolutionary algorithms (EAs) although boosting has been integrated with EAs in several other ways.

For example, boosting has been integrated with genetic algorithms (GAs) to solve classification problems [8]–[10]. The Boosting Genetic Algorithm integrates boosting with a GA to discover classification rules [8]. A GA is used as a base classifier in which each individual represents a classification rule. A boosting algorithm aggregates multiple base classifiers to build a more accurate classifier than them.

GPBoost [11] and its variants (e.g., [12]) integrate boosting with genetic programming (GP) to solve regression problems. A GP algorithm is used as a base learner (i.e., regression solver), and a boosting algorithm aggregates multiple base learners.

III. Quality Indicators

This section describes 15 representative quality indicators that the proposed boosting method uses.

A. Hypervolume Indicator (I_H)

I_H measures the volume of a hypercube that an individual dominates in the objective space [13]. The hypercube is formed with the individual and the reference point representing the highest (or worst) possible objective values $\vec{r} = (r_1, r_2, .., r_n)$ where n denotes the number of objectives. I_H of an individual $\vec{x}$ is calculated as follows where $f_i(\vec{x})$ denotes the i^{th} objective function value of $\vec{x}$.

$$I_H(\vec{x}) = \prod_{i=1}^{n} |r_i - f_i(\vec{x})| \tag{2}$$

I_H is intended to favor balanced individuals in objective space rather than extreme ones [13].

B. Weighted Hypervolume Indicator (I_{W1} to I_{W9})

I_W is an extension to I_H in that I_W places different weights on different regions in the objective space while I_H places the uniform weight on all regions [4]. I_W of an individual $\vec{x} = (x_1, x_2, ..., x_n)$ is computed as follows.

$$\begin{aligned} I_W(\vec{x}) &= \int_{(x_1,x_2,...,x_n)}^{(r_1,r_2,...,r_n)} w(\vec{a})dz \\ & where\ w(\vec{a}) = \frac{\sum_{i=1}^{n} e^{k_i(r_i - a_i)}}{\sum_{i=1}^{n} e^{k_i}} \end{aligned} \tag{3}$$

$w(\vec{a})$ denotes the weight of a point $\vec{a} = (a_1, a_2, ..., a_n)$ in the objective space. It is calculated by applying a weight distribution $\vec{k} = (k_1, k_2, .., k_n)$. k_i is the weight assigned to the i^{th} objective. Given a greater k_i value, I_W favors extreme individuals that are closer to the f_i axis in the objective space. Note that I_W is equal to I_H when $\vec{k} = (0, 0, .., 0)$.

As shown in Table I, this paper considers nine variants of I_W (I_{W1} to I_{W9}) based on nine different combinations of k_1 and k_2 values. Note that this papers uses a training problem whose objective space is two dimensional.

Table I: 9 Variants of the Weighted Hypervolume Indicator

I_W variants	k_1	k_2	I_W variants	k_1	k_2
I_{W1}	10	10	I_{W6}	0	20
I_{W2}	10	0	I_{W7}	30	30
I_{W3}	0	10	I_{W8}	30	0
I_{W4}	20	20	I_{W9}	0	30
I_{W5}	20	0			

C. HypE Indicator (I_{HypE})

I_{HypE} is also an extension to I_H. This indicator places different weights on different portions in the hypervolume that an individual dominates. The hypervolume is divided into multiple portions based on how many other individuals dominate it as well. I_{HypE} of $\vec{x}$ is computed as follows [5].

$$I_{HypE}(\vec{x}) = \sum_{i=1}^{\mu} \frac{1}{i} H_i(\vec{x}) \tag{4}$$

μ denotes the population size (i.e., the number of individuals in the population). $H_i(a)$ denotes the hypervolume that is dominated by $\vec{x}$ and other $(i-1)$ individuals in the population. H_1 is the hypervolume that $\vec{x}$ dominates exclusively. The highest weight of 1 is given to H_1. H_2 is the hypervolume that $\vec{x}$ and another individual dominate. The second highest weight of $\frac{1}{2}$ is given to H_2. The lowest weight of $\frac{1}{\mu}$ is given to H_μ, which all individuals in the population dominate.

D. Binary $\varepsilon+$ Indicator ($I_{\varepsilon+1}$ and $I_{\varepsilon+2}$)

I_ε takes two individuals ($\vec{x}$ and $\vec{y}$) and measures the distance between them on a per-objective basis. It is computed as follows [6].

$$I_{\epsilon+}(\vec{x}, \vec{y}) = max_{i \in \{1,..,n\}}(f_i(\vec{x}) - f_i(\vec{y})) \tag{5}$$

This paper considers two methods to evaluate the quality of an individual ($\vec{x}$) against the other individuals in the population P. The first method is to sum up binary indicator values.

$$I_{\epsilon+1}(\vec{x}) = \sum_{\vec{y} \in P \setminus \{\vec{x}\}} I_{\epsilon+}(\vec{y}, \vec{x}) \tag{6}$$

The second method amplifies the influence of dominating individuals over dominated one.

$$I_{\epsilon+2}(\vec{x}) = \sum_{\vec{y} \in P \setminus \{\vec{x}\}} -e^{-I_{\epsilon+}(\vec{y}, \vec{x})/l} \tag{7}$$

l is a scaling coefficient. $l = 0.05$ in this paper, which is a recommended value in [6].

E. Binary Hypervolume Indicator (I_{HD1} and I_{HD2})

I_{HD} takes two individuals ($\vec{x}$ and $\vec{y}$) and measures the hypervolume dominated by $\vec{x}$ but not by $\vec{y}$ [6].

$$I_{HD}(\vec{x},\vec{y}) = \begin{cases} H(\vec{x}) - H(\vec{y}) & \text{if } \vec{x} \text{ dominates } \vec{y} \\ H(\vec{x}) - H(\vec{x}) \cap H(\vec{y}) & \text{otherwise} \end{cases} \quad (8)$$

$H(\vec{x})$ denotes the hypervolume that $\vec{x}$ dominates.

Similar to $I_{\epsilon+1}$ and $I_{\epsilon+2}$, this paper considers two variants, I_{HD1} and I_{HD2}, to evaluate the quality of an individual ($\vec{x}$) against the other individuals in the population. $I_{HD1}(\vec{x})$ and $I_{HD2}(\vec{y})$ are computed by replacing $I_{\epsilon+}(\vec{y},\vec{x})$ with $I_{HD}(\vec{x},\vec{y})$ in Equations 6 and 7, respectively.

IV. Boosting Selection Operators

Algorithm 1 shows the proposed boosting process, which employs the AdaBoost algorithm [14]. It takes M indicator-based selection operators S and aggregates top T operators S^* ($T \leq M$). This paper uses 15 tournament selection operators that use 15 indicators described in Section III ($M = 15$). T aggregated operators have their weights: $W^* = \{\alpha_1, \alpha_2, ..., \alpha_T\}$.

The proposed boosting process is carried out through an offline training with a multiobjective optimization problem in which Pareto-optimal solutions are known. This training problem is used to generate N training populations, $\{p_1, p_2, .., p_N\}$, each of which contains μ individuals (Line 2). Of the μ individuals, N_p individuals are Pareto-optimal and $\mu - N_p$ individuals are randomly generated. Those N_p Pareto-optimal individuals are selected from a training problem so that they are equally distributed on the Pareto-optimal front. Each training population has a weight w_i ($1 \leq i \leq N$). Its initial value is $1/N$ (Line 3).

The proposed boosting process iteratively executes a loop (Line 4 to 15) T times and selects one operator in each iteration. (It selects T operators through T iterations.) In each iteration, each of M operators selects an individual N_p times (i.e., N_p individuals in total) on each training population (Line 5). The operator's individual selection is considered *successful* if it selects $N_p \times \theta$ or more Pareto-optimal individuals ($\theta < 1$). Given this condition, the selection error of each operator is calculated as shown in Line 7. The error is weighted with each training population's weight w_i ($1 \leq i \leq N$). Then, the proposed boosting process chooses the operator s_t^* that has the lowest selection error (Lines 8 and 9) and computes the operator's weight (Lines 10, 11 and 12). A lower selection error contributes to a higher weight.

Finally, each training population's weight is adjusted as shown in Lines 13 and 14. The weight decreases if s_t^*'s individual selection is successful; otherwise, it increases. This way, in subsequent loop iterations, the proposed boosting process focuses on the training populations on which individual selection failed and favors the operators that perform successful individual selection on those populations.

Algorithm 1 Boosting Selection Operators

Input: $S = \{s_1, s_2, .., s_M\}$, a set of M operators
Output: $S^* = \{s_1^*, s_2^*, .., s_T^*\}$, a set of T aggregated operators
Output: $W^* = \{\alpha_1, \alpha_2, ..., \alpha_T\}$, Weights of T aggregated operators
1: $S^* = \phi$, $W^* = \phi$
2: Generate N training populations: $\{p_1, p_2, .., p_N\}$
3: Initialize each training population's weight: $w_i(1) = 1/N, 1 \leq i \leq N$
4: **for** $t = 1$ to T **do**
5: Each operator s_j performs individual selection N_p times on each training population p_i.
6: Calculate the weighted selection error (e_j) for s_j
7: $e_j = \sum_{i=1}^{N} w_i I_{ji}$ where $I_{ji} = \begin{cases} 0 & \text{if } s_j\text{'s selection is successful on } p_i \\ 1 & \text{otherwise} \end{cases}$
8: Choose an operator s_t^* such that s_t^*=$\arg\min_{s_j \in S} e_j$
9: Add s_t^* to S^*
10: e_t^* = the weighted selection error of s_t^*
11: Calculate the weight (α_t) of s_t^* as $\alpha_t = \frac{1}{2} log\left(\frac{1-e_t^*}{e_t^*}\right)$ where e_t^* denotes the weighted selection error of s_t^*
12: Add α_t to W^*
13: Adjust w_i as $w_i(t+1)$ = $\begin{cases} w_i(t)e^{-\alpha_t} & \text{if } s_t\text{'s selection is successful} \\ w_i(t)e^{\alpha_t} & \text{otherwise} \end{cases}$
14: Normalize $w_i(t+1)$ as $w_i(t+1) = \frac{w_i(t+1)}{\sum_{q=1}^{N} w_q(t+1)}$
15: **end for**
16: **return** S^*, W^*

Algorithm 2 Boosted Selection Operator

Input: $S^* = \{s_1^*, s_2^*, .., s_T^*\}$, T aggregated operators
Input: $W^* = \{\alpha_1, \alpha_2, ..., \alpha_T\}$, Weights of T aggregated operators
Input: P, a population of μ individuals
Output: an individual to be used as a parent for crossover
1: Each of T operators selects one individual from the population P with a v-way tournament. In total, T individuals are selected: $\{x_1, x_2, ..., x_T\}$
2: Calculate the weight of each individual x_i as $\varphi_i = \sum_{t=1}^{T} \alpha_t I_{ti}$ where $I_{ti} = \begin{cases} 1 & \text{if } s_t \text{ selects } x_i \\ 0 & \text{otherwise} \end{cases}$
3: Calculate the selection probability of x_i as $\delta_i = \frac{\varphi_i}{\sum_{i=1}^{T} \varphi_i}$
4: Select an individual from $\{x_1, x_2, ..., x_T\}$ based on δ_i.

V. Boosted Parent Selection

A boosted selection operator is constructed with T operators S^* and their weights W^*, which Algorithm 1 produces. Algorithm 2 shows how a boosted operator works. In a boosted operator, each of T operators first selects one individual from the population P with a v-way tournament (Line 1). In a v-way tournament, an operator randomly draws two individuals from P and chooses a superior one

based on a quality indicator that the operator uses. A weight φ_i ($1 \leq i \leq T$) is assigned to each of selected T individuals with a prioritized voting by T operators (Line 2). Priorities are given to individuals based on the weights of operators ($\{\alpha_1, \alpha_2, ..., \alpha_T\}$). Finally, a boosted operator chooses one of T individuals as a parent by deriving individual selection probability δ_i from φ_i ($1 \leq i \leq T$) (Lines 3 and 4).

VI. Experimental Evaluation

This section evaluates the proposed boosting method by integrating a boosted selection operator with a well-known EMOA, called NSGA-II [15]. The proposed boosting method and NSGA-II are configured as shown in Table II. Experiments were conducted with jMetal [16]. Every experimental result is obtained with 20 independent experiments.

Table II: Algorithmic Configurations

Parameter	Value	Parameter	Value
M (Algo. 1)	15	v (Algo. 2)	2, 3, 4, 5 or 6
T (Algo. 1 and 2)	6	Max # of generations	200
N (Algo. 1)	5000	Crossover operator	SBX
μ (Algo. 1 and 2)	100	Crossover rate	0.9
N_p (Algo. 1)	20	Mutation operator	Polynomial
θ (Algo. 1)	0.6	Mutation rate	1/ μ

Table III shows the six indicators that the proposed boosting method chosen from 15 indicators in order to construct a boosted selection opererator. Note that this evaluation study uses $M = 15$ and $T = 6$ in Algorithms 1 and 2.

Table III: Aggregated Indicators

Indicator	Weight (α)	Indicator	Weight (α)
I_{HD2}	0.2435	I_{W7}	0.1163
$I_{\epsilon+1}$	0.1865	I_{W1}	0.0418
I_{HypE}	0.1420	I_{W4}	0.0250

A. Training and Test Problems

This evaluation study uses ZDT1 as a training problem. ZDT1 is the simplest problem in the ZDT family problems [17]. It has a convex Pareto-optimal front in a two dimensional objective space.

ZDT2, ZDT3 are ZDT4 are used to evaluate a boosted selection operator that aggregates the indicators shown in Table III. Each of the problems has a two dimensional objective space. ZDT2 is essentially same as ZDT1 in terms of problem design and complexity; however, it has a concave Pareto-optimal front. ZDT3 and ZDT4 are harder problems than ZDT1. ZDT3 has five discontiguous Pareto-optimal fronts. ZDT4 has a large number of (20^9) local optima.

DTLZ1 and DTLZ7 [18] are also used as test problems. Both are harder problems than ZDT1. They have three dimensional objective spaces. DTLZ1 has a single contiguous Pareto-optimal front and DTLZ7 has four discontiguous Pareto-optimal fronts.

B. Optimality and Diversity Analysis

This section evaluates the optimality and diversity of individuals with hypervolume ratio (HVR) [19]. HVR quantifies the optimality and diversity of non-dominated individuals. A higher HVR indicates that non-dominated individuals are closer to the Pareto-optimal front and more diverse in the objective space.

Table IV shows the average HVR values that seven algorithms yield at the last generation in each problem. I_B represents an EMOA that integrates NSGA-II with a boosted selection operator aggregating the six indicators listed in Table III. Each of the other six algorithms represents an EMOA that integrates NSGA-II with a selection operator based on a single indicator. For example, I_{HD2} represents an EMOA that integrates NSGA-II with an I_{HD2}-based selection operator. v indicates the size of a tournament in parent selection. In each problem, 2-way to 6-way tournament selections are examined. A bold number indicates the best result among seven algorithms on a per-row basis.

Table IV: Average HVR at the last (the 200th) generation

Problem	v	I_B	I_{HD2}	$I_{\varepsilon+1}$	I_{HypE}	I_{W7}	I_{W1}	I_{W4}
ZDT1	2	0.987292	0.987013	0.987142	**0.987981**	0.987211	0.987611	0.987187
	3	**0.98792**	0.983065	0.978066	0.987779	0.888314	0.883464	0.891662
	4	**0.988359**	0.957663	0.964091	0.987074	0.717391	0.711519	0.713685
	5	**0.988414**	0.960003	0.955519	0.986026	0.606751	0.57291	0.600777
	6	**0.988484**	0.958924	0.950242	0.985216	0.51941	0.482579	0.492156
ZDT2	2	0.9767	**0.97744**	0.974778	0.976514	0.969523	0.95584	0.961119
	3	0.977363	**0.977393**	0.935134	0.977289	0.854956	0.850348	0.854738
	4	0.977615	**0.977671**	0.933695	0.975612	0.70477	0.705306	0.722501
	5	**0.97847**	0.977542	0.936609	0.975074	0.672209	0.664869	0.681658
	6	0.976969	**0.977422**	0.924938	0.97114	0.636568	0.619629	0.635382
ZDT3	2	**0.991595**	0.987221	0.985653	0.989607	0.990329	0.988545	0.988114
	3	0.994079	0.99411	0.9426	**0.994947**	0.851847	0.842669	0.835347
	4	**0.994872**	0.993858	0.910609	0.98264	0.652831	0.682493	0.692975
	5	**0.99506**	0.989828	0.8672	0.991706	0.512317	0.546651	0.533493
	6	**0.995535**	0.984116	0.832858	0.988276	0.486224	0.476092	0.47217
ZDT4	2	0.923535	0.881758	0.915589	0.84696	0.973355	0.974579	**0.976871**
	3	**0.969551**	0.86502	0.953591	0.903897	0.873296	0.855642	0.869665
	4	**0.979587**	0.885057	0.950269	0.872678	0.656291	0.767591	0.609153
	5	**0.979602**	0.894803	0.941216	0.917018	0.559693	0.586559	0.60671
	6	**0.979984**	0.893172	0.94369	0.92175	0.572723	0.483191	0.481425
DTLZ1	2	0.963187	0.968643	0.8065	**0.973117**	0.919929	0.92741	0.891426
	3	0.946998	**0.963644**	0.868354	0.958693	0.340036	0.333922	0.373305
	4	0.964603	**0.976193**	0.777159	0.965573	0.23241	0.357037	0.330733
	5	**0.976716**	0.972869	0.760283	0.967659	0.236244	0.375746	0.350181
	6	0.930752	**0.974245**	0.752827	0.966478	0.239547	0.370695	0.274772
DTLZ7	2	0.972382	0.852836	0.989412	0.904304	**0.988395**	0.943761	0.97001
	3	**0.990347**	0.921616	0.969646	0.968427	0.680902	0.677735	0.716543
	4	**0.990547**	0.869581	0.932087	0.901701	0.585396	0.593865	0.578626
	5	**0.973147**	0.886484	0.87774	0.968013	0.560746	0.556716	0.558532
	6	**0.989858**	0.919655	0.805475	0.901431	0.555464	0.555143	0.568183

In ZDT1, except in the case of $v = 2$, the proposed boosted selection operator (I_B) outperforms the other operators. This is not surprising because I_B is constructed with ZDT1 as a training problem. However, a very similar observation can be made in ZDT3, ZDT4 and DTLZ7, which are harder problems than ZDT1. Table IV demonstrates that the proposed boosting process can work with a simple training problem and the boosted selection operator can effectively solve harder problems.

In ZDT2 and DTLZ1, I_{HD2} performs slightly better than I_B. (In fact, I_B ties I_{HD2} in ZDT2 if HVR values are truncated to two decimal places.) An important observation is that the performance of I_{HD2} is inconsistent among different problems. Although I_{HD2} works well in ZDT2 and DTLZ1, it's performance is marginal in ZDT4 and DTLZ7.

In ZDT4, I_{HD2} never yields the HVR measure of 9.0 or higher. Other indicators exhibit similar inconsistencies. For example, $I_{\epsilon+1}$ performs well in ZDT1 but poorly in DTLZ1. (It never yields the HVR measure of 0.87 or higher in DTLZ1.)

In contrast, I_B's performance is much more consistent among different problems. Its worst HVR is 0.92 in ZDT4 while I_{HD2}'s worst is 0.85, $I_{\epsilon+1}$'s is 0.75, I_{HypE}'s is 0.84, I_{W7}'s is 0.23, I_{W7}'s is 0.33, and I_{W4}'s is 0.27. This shows that I_B allows different indicator-based operators to complement with each other well. In summary, Table IV demonstrates that the proposed I_B performs better than, or equally to, existing indicator-based selection operators in HVR (i.e., in optimality and diversity) in all test problems except DTLZ1 and I_B is more robust and stable than existing operators under different characteristics in different problems.

C. Convergence Velocity Analysis

This section evaluates the convergence velocity of seven different algorithms with HVR (Tables V to X). Each of these tables shows the number of generations that each algorithm requires to achieve a given HVR value. In each problem, 2-way (i.e., binary) to 6-way tournament selections are examined. A bold number indicates the best result among seven algorithms on a per-row basis. For example, I_B requires 114 generations to achieve the HVR value of 0.9 with a three-way tournament in ZDT2 (Table VI). Its convergence velocity is the fastest among seven algorithms. A number in parentheses indicates the ratio of convergence velocity between I_B and another algorithm (i.e., how faster or slower an algorithm's convergence is against I_B). In Table VI, I_{HD2}'s convergence is 17% slower than I_B in the case of three-way tournament selection. The character "x" indicates that an algorithm cannot achieve a given HVR value by the last generation.

Tables V to X show that I_B's convergence velocity is faster than the others' in ZDT2, ZDT3, ZDT4 and DTLZ7. It is not the best but fairly acceptable in ZDT1 and DTLZ1. The weighted hypervolume indicators (I_{W7}, I_{W1} and I_{W4}) possess greater convergence velocity in many problems to achieve the HVR value of 0.5. However, they often encounter premature convergence and fail to achieve higher HVR values when $v \geq 3$. I_{HypE} and I_{HD2} yield the greatest convergence velocity in ZDT1 and DTLZ1, respectively. However, their convergence velocities are not consistent among different problems. I_{HypE}'s convergence velocity is marginal in other problems than ZDT1. I_{HD2}'s is marginal in other problems than DTLZ1.

In contrast, I_B never encounters premature convergence. (It never fails to achieve the HVR value of 0.9.) Its convergence velocity is more consistent among different problems. It is more robust and stable than exiting operators under different characteristics in different problems by allowing them to complement with each other.

Table V: HVR in ZDT1

v	HVR	I_B	I_{HD2}	$I_{\varepsilon+1}$	I_{HypE}	I_{W7}	I_{W1}	I_{W4}
2	0.5	37	37(1.0)	40(1.08)	**36(0.97)**	37(1.0)	**36(0.97)**	37(1.0)
	0.75	55	56(1.02)	58(1.05)	**52(0.95)**	55(1.0)	54(0.98)	55(1.0)
	0.9	75	79(1.05)	81(1.08)	**73(0.97)**	77(1.03)	76(1.01)	77(1.03)
3	0.5	**34**	36(1.06)	40(1.18)	33(0.97)	38(1.12)	36(1.06)	37(1.09)
	0.75	51	53(1.04)	58(1.14)	**48(0.94)**	62(1.22)	61(1.2)	61(1.2)
	0.9	70	74(1.06)	87(1.24)	**67(0.96)**	x	x	x
4	0.5	**33**	**33(1.0)**	46(1.39)	34(1.03)	41(1.24)	41(1.24)	42(1.27)
	0.75	**49**	50(1.02)	64(1.31)	**49(1.0)**	x	x	x
	0.9	**67**	79(1.18)	93(1.39)	69(1.03)	x	x	x
5	0.5	**32**	35(1.09)	46(1.44)	32(1.0)	47(1.47)	55(1.72)	48(1.5)
	0.75	48	51(1.06)	64(1.33)	**47(0.98)**	x	x	x
	0.9	67	77(1.15)	96(1.43)	**64(0.96)**	x	x	x
6	0.5	34	33(0.97)	45(1.32)	**31(0.91)**	78(2.29)	x	x
	0.75	49	49(1.0)	66(1.35)	**46(0.94)**	x	x	x
	0.9	68	78(1.15)	93(1.37)	**63(0.93)**	x	x	x

Table VI: HVR in ZDT2

v	HVR	I_B	I_{HD2}	$I_{\varepsilon+1}$	I_{HypE}	I_{W7}	I_{W1}	I_{W4}
2	0.5	103	116(1.13)	88(0.85)	110(1.07)	62(0.6)	**61(0.59)**	64(0.62)
	0.75	123	135(1.1)	114(0.93)	126(1.02)	**79(0.64)**	82(0.67)	85(0.69)
	0.9	144	158(1.1)	134(0.93)	144(1.0)	**110(0.76)**	120(0.83)	119(0.83)
3	0.5	83	99(1.19)	91(1.1)	92(1.11)	**55(0.66)**	57(0.69)	57(0.69)
	0.75	99	117(1.18)	111(1.12)	106(1.07)	**82(0.83)**	84(0.85)	84(0.85)
	0.9	**114**	133(1.17)	134(1.18)	119(1.04)	x	x	x
4	0.5	76	97(1.28)	100(1.32)	96(1.26)	59(0.78)	59(0.78)	**56(0.74)**
	0.75	**90**	109(1.21)	125(1.39)	105(1.17)	x	x	x
	0.9	**108**	119(1.1)	143(1.32)	122(1.13)	x	x	x
5	0.5	80	94(1.18)	98(1.23)	94(1.18)	**59(0.74)**	60(0.75)	**59(0.74)**
	0.75	**95**	106(1.12)	128(1.35)	109(1.15)	x	x	x
	0.9	**109**	121(1.11)	163(1.5)	128(1.17)	x	x	x
6	0.5	73	91(1.25)	118(1.62)	91(1.25)	**61(0.84)**	65(0.89)	66(0.9)
	0.75	**87**	105(1.21)	133(1.53)	110(1.26)	x	x	x
	0.9	**102**	116(1.14)	151(1.48)	148(1.45)	x	x	x

Table VII: HVR in ZDT3

v	HVR	I_B	I_{HD2}	$I_{\varepsilon+1}$	I_{HypE}	I_{W7}	I_{W1}	I_{W4}
2	0.5	103	116(1.13)	88(0.85)	110(1.07)	62(0.6)	**61(0.59)**	64(0.62)
	0.75	123	135(1.1)	114(0.93)	126(1.02)	79(0.64)	**82(0.67)**	85(0.69)
	0.9	144	158(1.1)	134(0.93)	144(1.0)	**110(0.76)**	120(0.83)	119(0.83)
3	0.5	83	99(1.19)	91(1.1)	92(1.11)	**55(0.66)**	57(0.69)	57(0.69)
	0.75	99	117(1.18)	111(1.12)	106(1.07)	**82(0.83)**	84(0.85)	84(0.85)
	0.9	**114**	133(1.17)	134(1.18)	119(1.04)	x	x	x
4	0.5	76	97(1.28)	100(1.32)	96(1.26)	59(0.78)	59(0.78)	**56(0.74)**
	0.75	**90**	109(1.21)	125(1.39)	105(1.17)	x	x	x
	0.9	**108**	119(1.1)	143(1.32)	122(1.13)	x	x	x
5	0.5	80	94(1.18)	98(1.23)	94(1.18)	**59(0.74)**	60(0.75)	**59(0.74)**
	0.75	**95**	106(1.12)	128(1.35)	109(1.15)	x	x	x
	0.9	**109**	121(1.11)	163(1.5)	128(1.17)	x	x	x
6	0.5	73	91(1.25)	118(1.62)	91(1.25)	**61(0.84)**	65(0.89)	66(0.9)
	0.75	**87**	105(1.21)	133(1.53)	110(1.26)	x	x	x
	0.9	**102**	116(1.14)	151(1.48)	148(1.45)	x	x	x

VII. Conclusions

This paper proposes and evaluates a novel method that leverages a boosting algorithm to obtain an aggregated selection operator from various existing indicator-based selection operators. Experimental results show that a boosted selection operator outperforms exiting ones in optimality, diversity and convergence velocity. The proposed boosting process can work with a simple training problem, and the boosted operator can effectively solve harder problems. The boosted operator also exhibits robustness against different characteristics in different problems and yields stable performance.

Table VIII: HVR in ZDT4

v	HVR	I_B	I_{HD2}	$I_{\varepsilon+1}$	I_{HypE}	I_{W7}	I_{W1}	I_{W4}
2	0.5	121	129(1.07)	132(1.09)	136(1.12)	98(0.81)	**95(0.79)**	**95(0.79)**
	0.75	148	152(1.03)	158(1.07)	177(1.2)	119(0.8)	**113(0.76)**	114(0.77)
	0.9	191	210(1.1)	193(1.01)	222(1.16)	144(0.75)	137(0.72)	**135(0.71)**
3	0.5	106	114(1.08)	111(1.05)	118(1.11)	83(0.78)	81(0.76)	**80(0.75)**
	0.75	129	142(1.1)	146(1.13)	148(1.15)	107(0.83)	**104(0.81)**	109(0.84)
	0.9	**158**	238(1.51)	178(1.13)	198(1.25)	x	x	x
4	0.5	94	99(1.05)	108(1.15)	107(1.14)	84(0.89)	**71(0.76)**	94(1.0)
	0.75	**115**	136(1.18)	138(1.2)	153(1.33)	x	141(1.23)	x
	0.9	**141**	235(1.67)	175(1.24)	225(1.6)	x	x	x
5	0.5	87	96(1.1)	108(1.24)	103(1.18)	92(1.06)	79(0.91)	**75(0.86)**
	0.75	**108**	131(1.21)	141(1.31)	130(1.2)	x	x	x
	0.9	**135**	217(1.61)	177(1.31)	170(1.26)	x	x	x
6	0.5	84	101(1.2)	100(1.19)	99(1.18)	**77(0.92)**	x	x
	0.75	**99**	131(1.32)	137(1.38)	136(1.37)	x	x	x
	0.9	**129**	211(1.64)	184(1.43)	190(1.47)	x	x	x

Table IX: HVR in DTLZ1

v	HVR	I_B	I_{HD2}	$I_{\varepsilon+1}$	I_{HypE}	I_{W7}	I_{W1}	I_{W4}
2	0.5	129	126(0.98)	132(1.02)	**110(0.85)**	157(1.22)	142(1.1)	155(1.2)
	0.75	153	150(0.98)	169(1.1)	**128(0.84)**	175(1.14)	163(1.07)	178(1.16)
	0.9	175	176(1.01)	224(1.28)	**157(0.9)**	194(1.11)	187(1.07)	204(1.17)
3	0.5	122	117(0.96)	127(1.04)	**111(0.91)**	x	x	245(2.01)
	0.75	139	**138(0.99)**	178(1.28)	140(1.01)	x	x	x
	0.9	**165**	180(1.09)	218(1.32)	172(1.04)	x	x	x
4	0.5	115	**105(0.91)**	141(1.23)	113(0.98)	x	x	x
	0.75	142	**133(0.94)**	183(1.29)	135(0.95)	x	x	x
	0.9	184	**154(0.84)**	x	162(0.88)	x	x	x
5	0.5	114	**100(0.88)**	148(1.3)	104(0.91)	x	x	x
	0.75	144	**121(0.84)**	195(1.35)	129(0.9)	x	x	x
	0.9	158	**132(0.84)**	x	159(1.01)	x	x	x
6	0.5	119	**96(0.81)**	124(1.04)	103(0.87)	x	x	x
	0.75	138	**118(0.86)**	197(1.43)	126(0.91)	x	x	x
	0.9	164	142(0.87)	x	**136(0.83)**	x	x	x

REFERENCES

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

[2] C. C. Coello, "Evolutionary multi-objective optimization: Some current research trends and topics that remain to be explored," *Front. Computat. Sci. China*, vol. 3, no. 1, 2009.

[3] T. Wagner, N. Beume, and B. Naujoks, "Pareto-, aggregation-, and indicator-based methods in many-objective optimization," in *Proc. Int'l Conf. Evol. Multi-criterion Optimization*, 2007.

Table X: HVR in DTLZ7

v	HVR	I_B	I_{HD2}	$I_{\varepsilon+1}$	I_{HypE}	I_{W7}	I_{W1}	I_{W4}
2	0.5	49	52(1.06)	52(1.06)	49(1.0)	**48(0.98)**	52(1.06)	49(1.0)
	0.75	**61**	71(1.16)	64(1.05)	64(1.05)	66(1.08)	71(1.16)	66(1.08)
	0.9	**83**	x	84(1.01)	152(1.83)	96(1.16)	130(1.57)	104(1.25)
3	0.5	44	43(0.98)	50(1.14)	**42(0.95)**	57(1.3)	53(1.2)	52(1.18)
	0.75	55	55(1.0)	66(1.2)	**54(0.98)**	x	x	x
	0.9	**71**	91(1.28)	93(1.31)	75(1.06)	x	x	x
4	0.5	42	43(1.02)	50(1.19)	**40(0.95)**	60(1.43)	59(1.4)	64(1.52)
	0.75	**52**	57(1.1)	69(1.33)	54(1.04)	x	x	x
	0.9	**68**	x	118(1.74)	179(2.63)	x	x	x
5	0.5	40	**38(0.95)**	51(1.27)	38(0.95)	70(1.75)	73(1.83)	67(1.68)
	0.75	51	52(1.02)	76(1.49)	**48(0.94)**	x	x	x
	0.9	**68**	x	x	69(1.01)	x	x	x
6	0.5	39	**38(0.97)**	52(1.33)	39(1.0)	74(1.9)	73(1.87)	66(1.69)
	0.75	**49**	51(1.04)	91(1.86)	54(1.1)	x	x	x
	0.9	**64**	82(1.28)	x	174(2.72)	x	x	x

[4] E. Zitzler, D. Brockho, and L. Thiele, "The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicators Via Weighted Integration," in *Proc. of Int'l Conference on Evolutionary Multi-Criterion Optimization*, 2007.

[5] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Computat.*, vol. 19, no. 1, 2011.

[6] E. Zitzler and S. Kuenzli., "Indicator-based selection in multiobjective search," in *Proc. of Int'l Conference on Parallel Problem Solving from Nature*, 2004.

[7] P. Boonma and J. Suzuki, "Prospect indicator based evolutionary multiobjective optimization algorithm," in *Proc. IEEE Congress on Evolutionary Computation*, 2011.

[8] B. Liu, M. B., and A. H.A., "Improving genetic classifiers with a boosting algorithm," in *Proc. IEEE Congress on Evolutionary Computation*, 2003.

[9] B. Liu, B. McKay, and H. A. Abbass, "Feature selection combining genetic algorithm and adaboost classifiers," in *Proc. Int'l Conference on Pattern Recognition*, 2008.

[10] I. Yalabik and T.-V. Fatos, "A pattern classification approach for boosting with genetic algorithms," in *Proc. Int'l Symposium on Computer and Information Sciences*, 2007.

[11] G. Paris, D. Robilliard, and C. Fonlupt, "Applying boosting techniques to genetic programming," in *Proc. European Conference on Artificial Evolution*, 2002.

[12] L. V. Souza, A. R. T. Pozo, J. C. M. Da Rosa, and C. Neto, "Technique using correlation coefficient to improve time series forecasting accuracy," in *Proc. IEEE Congress on Evolutionary Computation*, 2007.

[13] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms: A comparative study," in *Proc. Int'l Conference on Parallel Problem Solving from Nature*, 1998.

[14] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, 1997.

[15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans Evol. Computat.*, vol. 6, no. 2, 2002.

[16] 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.

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

[18] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization*, A. Abraham, R. Jain, and R. Goldberg, Eds. Springer, 2005.

[19] D. A. V. Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm test suites," in *Proc. ACM Symposium on Applied Computing*, 1999.