An Adaptive Recombination-Based Extension of the iMOACO$_{\mathbb{R}}$ Algorithm

Ashraf M. Abdelbar
Dept. of Math. & Comp. Sci.
Brandon University
Manitoba, Canada
Email: abdelbara@brandonu.ca

Khalid M. Salama
School of Computing
University of Kent
Canterbury, United Kingdom
Email: kms39@kent.ac.uk

Jesús Guillermo Falcón-Cardona, Carlos A. Coello Coello
CINVESTAV-IPN
Evolutionary Computation Group
Mexico City, Mexico
Email: jfalcon@computation.cs.cinvestav.mx,
ccoello@cs.cinvestav.mx

***Abstract*—Commonly, Ant Colony Optimization algorithms have been applied to the solution of single- and multi-objective optimization problems (MOPs). However, in recent years, a number of approaches have been proposed to solve problems with continuous search spaces. One remarkable proposal is the indicator-based Multi-Objective Ant Colony Optimizer for continuous search spaces (iMOACO$_{\mathbb{R}}$) which is based on the ACO$_{\mathbb{R}}$ algorithm and the $R2$ performance indicator, aiming to solve continuous many-objective optimization problems (i.e., MOPs having more than three objective functions). In previous work, we presented an extension of iMOACO$_{\mathbb{R}}$, called iMOACO$_{\mathbb{R}}$-R, in which a recombination operator is employed for solution construction with a fixed, externally-specified probability. In the present work, we introduce a further adaptive variation, called iMOACO$_{\mathbb{R}}$-AR, in which the frequency of applying recombination is dynamically adapted based on the recent past performance of the recombination operator. Our proposal is compared to iMOACO$_{\mathbb{R}}$ and iMOACO$_{\mathbb{R}}$-R using 64 standard problems from the multi-objective optimization literature with number of objectives ranging from 3 to 10. Experimental results show that iMOACO$_{\mathbb{R}}$-AR outperforms iMOACO$_{\mathbb{R}}$ and iMOACO$_{\mathbb{R}}$-R in most of the test problems.**

I. Introduction

The constant study of animals and insects and their social structures has been a promising source for the design of methods applied in the engineering, industrial and scientific fields. Researchers have observed that simple actions made by individuals (e.g., ants, birds, bacteria) encourage the development of a more sophisticated global behavior [1]. In some cases, the global behavior exhibits interesting optimization properties. One outstanding example is related to ant colonies whose foraging behavior tends to find the shortest path from the nest to a source of food by using pheromone as an indirect communication method. Ant Colony Optimization (ACO) [2] is a metaheuristic that draws ideas from real ants in order to solve hard optimization problems, mainly combinatorial ones.

Real-world problems often involve the simultaneous optimization of several objective functions which are mutually in conflict. These problems are the so-called multi-objective optimization problems (MOPs). Due to the inner conflict, the solution of a MOP is a set of points that represent the best possible trade-offs among the objectives. Such solutions cannot be improved in one objective without being worsened in another one. Different mathematical programming techniques have been proposed for tackling MOPs [3]. However, these methods present numerous limitations (e.g., continuity of the objective functions is required, most of them generate a single solution per execution). Consequently, Multi-Objective Evolutionary Algorithms, that are population-oriented methods based on the principles of natural selection, have been proposed to solve these highly complex problems [4]. However, other bio-inspired metaheuristics such as ACO have been also applied to solve MOPs.

In recent years, the ACO metaheuristic has been extended to solve single-objective optimization problems with continuous search spaces. To the authors best knowledge, ACO$_{\mathbb{R}}$ [5] is currently the best ACO-based algorithm for these kind of problems. The underlying idea of ACO$_{\mathbb{R}}$ is the use of a pheromone structure, similar to an archive, that keeps track of the best solutions found so far. Moreover, the decision space is modeled by means of multimodal Gaussian functions. In 2017, Falcón-Cardona and Coello Coello [6] introduced iMOACO$_{\mathbb{R}}$, an extension of ACO$_{\mathbb{R}}$, that carries out multi-objective optimization [4] in continuous search spaces, and is designed specifically for problems with four or more objectives because of the use of a quality indicator to increment the selection pressure.

In previous work [7], we introduced a variation, iMOACO$_{\mathbb{R}}$-**R**, that incorporates recombination of solutions from the iMOACO$_{\mathbb{R}}$ solution archive, with the probability of deployment of recombination determined by a fixed, externally specified probability P_r. In the present work, we present a further adaptive extension, called iMOACO$_{\mathbb{R}}$-**AR**, in which the frequency of deployment of recombination is dynamically adapted during the algorithm's execution, instead of being determined by a fixed probability P_r. We evaluate our proposal using 64 problems from the DTLZ [8] and WFG [9] test suites, with number of objectives ranging from 3 to 10, and use the hypervolume indicator to assess performance. Our experimental results show that iMOACO$_{\mathbb{R}}$-**AR** outperforms iMOACO$_{\mathbb{R}}$ and iMOACO$_{\mathbb{R}}$-**R** in most of the test problems.

The remainder of this paper is organized as follows. Section II introduces the mathematical background of multi-objective optimization. An introduction to iMOACO$_{\mathbb{R}}$ is briefly sketched in Sect. III. The proposed approach is outlined in Sect. IV. Section V is devoted to establish the experiment methodology,

while the results are presented in Sect. VI. Finally, Sect. VII concludes with some final remarks.

II. Preliminaries

The general multi-objective optimization problem[1] is formulated as:

$$\min_{\vec{x}\in\mathbb{R}^n} \vec{F}(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_m(\vec{x})]^T \tag{1}$$

subject to:

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, \ldots, k \tag{2}$$

$$h_j(\vec{x}) = 0 \quad j = 1, 2, \ldots, p \tag{3}$$

where $\vec{x} = (x_1, x_2, \ldots, x_n)^T$ is the n-dimensional vector of decision variables; $f_i : \mathbb{R}^n \mapsto \mathbb{R}$, $i = 1, \ldots, m$ are the objective functions and $g_i, h_j : \mathbb{R}^n \mapsto \mathbb{R}$, $i = 1, \ldots, k$, $j = 1, \ldots, p$ are the constraint functions of the problem which define that feasible region Ω.

When solving a MOP, the aim is to find in Ω a subset of solutions $\vec{x}^*$ that yield the optimum values of all objective functions. In other words, the particular set that represents the best possible trade-offs among the objective functions. In furtherance of determining which solutions are optimal, the most common order relation used in multi-objective optimization is the Pareto dominance relation.

Definition 1 (Pareto Dominance): Given two vectors $\vec{u}, \vec{v} \in \mathbb{R}^n$, we say that $\vec{u}$ **dominates** $\vec{v}$ (denoted by $\vec{u} \prec \vec{v}$) if $u_i \leq v_i$ for $i = 1, \ldots, n$ and there exists at least an index $j \in \{1, \ldots, m\}$ such that $u_j < v_j$.

Definition 2 (Weak Pareto Dominance): Given two vectors $\vec{u}, \vec{v} \in \mathbb{R}^n$, we say that $\vec{u}$ **weakly dominates** $\vec{v}$ (denoted by $\vec{u} \preceq \vec{v}$) if $u_i \leq v_i$ for all $i = 1, \ldots, n$.

Definition 3 (Pareto Optimality): We say that a vector of decision variables $\vec{x}^* \in \Omega$ is **Pareto optimal** if there does not exist another $\vec{x} \in \Omega$ such that $\vec{F}(\vec{x}) \prec \vec{F}(\vec{x^*})$.

Definition 4: The **Pareto Optimal Set** $\mathcal{P}^*$ is defined by:

$$\mathcal{P}^* = \{\vec{x}^* \in \Omega \mid \vec{x} \text{ is Pareto optimal}\}$$

Definition 5: The **Pareto Front** $\mathcal{PF}^*$ is defined by:

$$\mathcal{PF}^* = \{\vec{F}(\vec{x}^*) \in \mathbb{R}^m \mid \vec{x}^* \in \mathcal{P}^*\}$$

The ideal objective vector and the nadir objective vector are two especial reference points that bound $\mathcal{PF}^*$.

Definition 6: The *Ideal Objective Vector* ($\vec{z}^* \in \mathbb{R}^m$) is constructed using the minimum of each of the objective functions, considered separately. Each ith-component of the ideal vector is defined as $z_i^* = \min_{\vec{x}} f_i(\vec{x})$.

[1]Without loss of generality, we will assume only unconstrained minimization problems. To transform a minimization problem into a maximization one, we can use: $\max f = -\min(-f)$

Definition 7: The *Nadir Objective Vector* ($\vec{z}^{nad} \in \mathbb{R}^m$) is constructed using the worst values of $\mathcal{PF}^*$. Each ith-component is defined as $z_i^{nad} = \max_{\vec{x}\in\mathcal{P}^*} f_i(\vec{x})$.

In order to assess the perfomance of MOEAs, a wide variety of quality indicators have been proposed in the specialized literature [10]. Let $\mathcal{A}$ be a finite set of points that approximate $\mathcal{PF}^*$ and Ψ be the set of all approximation sets. Mathematically, a k-ary quality indicator I is a function $I : \Psi^k \to \mathbb{R}$, which assigns to each vector $(\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_k)$ of k approximate Pareto fronts a real value $I(\mathcal{A}_1, \cdots, \mathcal{A}_k)$.

Definition 8 (Unary $R2$ indicator): The unary $R2$ indicator is defined as follows:

$$R2(\mathcal{A}, W) = -\frac{1}{|W|} \sum_{\vec{w}\in W} \max_{\vec{a}\in\mathcal{A}} \{u_{\vec{w}}(\vec{a})\}, \tag{4}$$

where W is a set of m-dimensional weight vectors and $u_{\vec{w}} : \mathbb{R}^m \mapsto \mathbb{R}$ is a scalarizing function, parameterized by $\vec{w} \in W$, that assigns a real value to each solution vector.

III. Previous related work

In 2017, Falcón-Cardona and Coello proposed the indicator-based Many-Objective Ant Colony Optimizer for continuous search spaces (iMOACO$_\mathbb{R}$) [6] based on the ACO$_\mathbb{R}$ [5] search engine. The most important element of every ACO-based algorithm is the design of the pheromone matrix since it stores knowledge through the search process to solve the optimization problem [2]. The pheromone matrix of ACO$_\mathbb{R}$ stores the best N solutions found so far and it sorts them according to the quality of the objective function. However, this scheme cannot be directly implemented in iMOACO$_\mathbb{R}$ since the Pareto dominance does not establish a total order. Hence, Falcón-Cardona and Coello proposed to use the $R2$ indicator [11] to transform the multi-objective problem into a single-objective one and, thus, impose a total order. For this purpose, the $R2$-ranking algorithm [12] was employed to rank the population in a similar fashion to the nondominated sorting [13] and, then, store the best N solutions according to the rank assigned. Figure 1 shows the pheromone matrix of iMOACO$_\mathbb{R}$. For each solution $\vec{x}^j, j = 1, \ldots, N$, auxiliary fields store its vector of objective values, the rank assigned and a weight value w_j. Moreover, all k^{th} components of the N solutions are employed to define a Gaussian-Kernel Probability Density Function G^k as follows:

$$G^k(y) = \sum_{j=1}^{N} w_j g_j^k(y) = \sum_{j=1}^{N} w_j \frac{1}{\sigma_k^j \sqrt{2\pi}} e^{-\frac{\left(y-\mu_k^j\right)^2}{2{\sigma_k^j}^2}}, \tag{5}$$

where $k = 1, \ldots, n$ and $G^k(y)$ depends on three parameter vectors: $\vec{w}$ is the vector of weights associated with the individual Gaussian functions, $\vec{\mu}_k$ is the vector of means, and $\vec{\sigma}_k$ is the vector of standard deviations. $\vec{\mu}_k = \{\mu_k^1, \mu_k^2 \ldots, \mu_k^N\} = \{x_k^1, x_k^2, \ldots, x_k^N\}$, and each $\sigma_k^j \in \vec{\sigma}_k$ is computed as follows:

$$\sigma_k^j = \xi \sum_{l=1}^{N} \frac{|x_k^l - x_k^j|}{N-1} \tag{6}$$

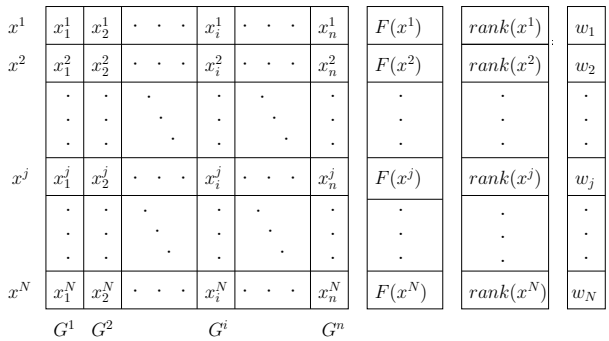

Fig. 1. Pheromone matrix employed by iMOACO$_{\mathbb{R}}$. The solutions are sorted according to its rank generated by the $R2$-ranking algorithm, where 1 is the better rank. Ranks can be repeated.

where $\xi > 0$ is a parameter that controls the convergence rate, simulating the evaporation of pheromones.

At each iteration, the weights $w_j, j = 1, \ldots, N$ are computed using the following formula:

$$w_j = \frac{1}{qN\sqrt{2\pi}} e^{-\frac{\left(rank\left(x^j\right)-1\right)^2}{2q^2N^2}} \tag{7}$$

where $q > 0$ is a parameter that controls the diversification process of the search. It is worth noting that q and ξ controls exploration and exploitation, respectively. Then, each of the M ants performs n construction steps to create a new solution $\vec{x}^{\text{new}}$, where each component x_k^{new} is drawn by sampling the rth Gaussian function that is part of G^k. The index $r \in \{1, \ldots, N\}$ is selected with probability

$$\Pr(\text{select } r) = \frac{w_r}{\sum_{l=1}^{N} w_l}. \tag{8}$$

Finally, the M newly created solutions compete with the ones in the pheromone matrix to be part of the pheromone matrix in the next iteration. Hence, iMOACO$_{\mathbb{R}}$ implements a $(\mu + \lambda)$ selection scheme, where $\mu = N$ and $\lambda = M$, using the $R2$-ranking algorithm [12].

IV. Proposed Approach: Incorporating Recombination

Recombination is a standard component of Evolutionary Algorithms (EA), and its use within discrete ACO models has been explored [14]. In this paper, we introduce iMOACO$_{\mathbb{R}}$-**AR**, an extension of iMOACO$_{\mathbb{R}}$ in which solutions are constructed either by recombination of members of the solution archive or by ACO$_{\mathbb{R}}$'s usual solution construction mechanism (Eqs. 5-8), with the relative frequency of applying recombination dynamically adapted during the algorithm's execution. In previous work [7], we proposed iMOACO$_{\mathbb{R}}$-**R**, an earlier non-adaptive variation in which recombination was deployed with a fixed externally-specified probability P_r. In iMOACO$_{\mathbb{R}}$-**R**, the w-weights are calculated at the start of each iteration according to Eq. (7) in the usual way, and each ant starts each solution construction step by applying Eq. (8) in the usual way. Let $\vec{x}^a$ refer to the solution selected according to Eq. (8). With probability $(1 - P_r)$, Eqs. (5-6) are then applied in the usual way to complete solution construction.

On the other hand, with probability P_r, recombination is applied: a second solution $\vec{x}^b$ is randomly selected from the archive with a uniform distribution, and a recombination operator is applied to $\vec{x}^a$ and $\vec{x}^b$ to produce an offspring solution $\vec{x}^c$. At present, we use uniform crossover (also known as N-point crossover) as the recombination operator, although other recombination operators can also be used. Uniform crossover means that for two N-dimensional solutions $\vec{x}^a$ and $\vec{x}^b$, the offspring $\vec{x}^c$ is produced as follows:

- For each $i = 1, \ldots, N$:
  - x_i^c takes the value of x_i^a with probability 0.5, and otherwise takes the value of x_i^b.

Finally, after each ant has constructed its solution, the newly created solutions compete for a place in the archive against the existing members of the population.

In iMOACO$_{\mathbb{R}}$-**AR** algorithm, solution construction is made by one of the following modes:

1) mode m_1 represents ACO$_{\mathbb{R}}$'s standard solution construction mechanism,
2) mode m_2 represents recombination.

Every member of the archive has an associated additional field ϕ which records which mode was used to generate that solution. When the solution archive is first initialized with random solutions at the start of the computation, the ϕ field is initialized to mode m_1 for all solutions. In other words, the initial random solutions are treated as if they were constructed using ACO$_{\mathbb{R}}$'s standard solution construction mechanism (represented by m_1).

The relative utility of each of the two modes of solution construction can be indirectly assessed by the number (and relative rank) of elements in the population constructed by that mode. The idea is that good solutions will survive for longer in the archive before being displaced, and will occupy a higher place (lower rank) in the archive. The first step in solution construction is to probabilistically select a mode. This decision is made probabilistically based on the relative representation of each mode in the population archive. Let the set of modes be denoted M, where

$$M = \{m_1, m_2\} \tag{9}$$

The utility u of a mode $m \in M$ is defined as:

$$u(m) = \sum_{\vec{x}^j \in A, \phi(\vec{x}^j) = m} \left[N - rank(\vec{x}^j)\right] \tag{10}$$

where A denotes the solution archive, N is the number of solutions in the archive, $rank(\vec{x}^j)$ denotes the rank of $\vec{x}^j$ (with a rank of 1 representing the best archived solution). Thus, $u(m)$ is the sum of the rank-complements of all solutions in the archive whose ϕ field is equal to m. A solution with a rank of 1 (the best solution in the archive) will have a rank-complement of $N - 1$, and a solution with a rank of N (the

worst solution) will have a rank-complement of 0. Intuitively, $u(m)$ is a measure of the representation of m in the archive. Note that in general (ignoring the effect of repeated ranks):

$$u(m_1) + u(m_2) = \sum_{i=0}^{N-1} i \equiv \frac{1}{2}(N-1)N \tag{11}$$

We would like the probability of selecting each mode $m \in M$ to be proportional to its utility, which would suggest the following simple roulette wheel formula:

$$\Pr(\text{select } m) = \frac{u(m)}{\sum_{s \in M} u(s)} \tag{12}$$

However, this formula has the drawback that the probability of selecting a mode can be a hard-zero, which is of course undesirable. Therefore, instead of Eq. (12), we adopt the following formula:

$$\Pr(\text{select } m) = \frac{u(m) + eN}{\sum_{s \in M} [u(s) + eN]} \tag{13}$$

where $e > 0$ is a constant. The term eN in the numerator and denominator is meant to play a role similar to a Laplace correction, i.e. to ensure that each mode can never die out completely. In the present work, we use $e = 1$. If the population was entirely generated by a given mode m (as is the case in the initial randomly-generated population, which is treated as if it were entirely constructed using m_1), then the probability of selection of the other mode m' will still be non-zero. In fact, the probability of selection of the other mode m' will generally be (again, ignoring the effect of repeated-ranks):

$$\begin{aligned} \Pr(\text{select } m') &= \frac{N}{\frac{1}{2}(N-1)N + 2N} \\ &= \frac{1}{\frac{1}{2}(N-1) + 2} = \frac{2}{N+3} \approx \frac{2}{N} \end{aligned} \tag{14}$$

which is small but still non-negligible.

Because the initial randomly-generated population is initialized to appear as if it were entirely constructed using m_1, the probability of selection of m_2 in the very first iteration will be as shown in Eq. (14). If the number of ants M is set equal to N, which is a common setting in iMOACO$_\mathbb{R}$, then mode m_2 will be selected about twice in the first iteration. The number of solutions constructed by m_2 (recombination) in the early iterations will therefore start small. If those solutions are poor and fail to secure a place in the population, or even secure a poorly-ranked position in the population, then $\Pr(m_2)$ will remain small. On the other hand, suppose that at a particular point in the computation, recombination produces solutions that are of a higher quality relative to those produced by m_1. Such m_2-produced solutions will secure well-ranked positions in the archive, and $\Pr(m_2)$ will gradually increase. Furthermore, depending on the problem, if there are regions of the search space for which one mode is better suited than the other, then $\Pr(m_1)$ and $\Pr(m_2)$ can fluctuate relative to each other as the algorithm progresses through the search space.

Thus, to summarize, iMOACO$_\mathbb{R}$-**AR** differs from iMOACO$_\mathbb{R}$ as follows:

1) Each element in the population has an additional field ϕ which stores the mode that was used to construct that solution: either m_1 (denoting ACO$_\mathbb{R}$'s usual solution construction mechanism) or m_2 (denoting recombination). For the initial randomly generated population, the ϕ field is initialized to m_1 for all population elements.
2) At the start of each iteration, the utility of each of the two modes is calculated using Eq. (10).
3) The probability of selecting each mode is calculated according to Eq. (13), and is not re-calculated until all M ants have constructed their solutions and the next iteration is about to start.
4) When each of the M ants starts to construct a solution, it probabilistically chooses a mode according to the probabilities calculated at the start of the iteration.
5) If mode m_1 is selected, then Eqs. (5-6) are applied to construct a solution in the usual ACO$_\mathbb{R}$ way.
6) On the other hand, if mode m_2 is selected, then one solution $\vec{x}^a$ is selected according to Eq. (8). Another solution $\vec{x}^b$ is selected from the archive with a uniform distribution. Uniform crossover is applied to $\vec{x}^a$ and $\vec{x}^b$ to produce a new solution $\vec{x}^c$. The new solution $\vec{x}^c$ becomes one of the M newly constructed solutions that will compete for a place in the solution archive.

This adaptive approach, iMOACO$_\mathbb{R}$-**AR**, has several advantages over iMOACO$_\mathbb{R}$-**R**:

1) There is no longer the need for a P_r parameter. The frequency of deploying recombination is determined based on Eq. (13).
2) A search process will typically go through different phases. Different solution construction mechanisms will be better suited for different phases of the search, but it is of course not feasible to manually decide which mechanism is better for any given point in the search. iMOACO$_\mathbb{R}$-**AR** allows the mode that is more successful at a given point in the search process to be deployed more often, without any intervention from the user.

V. Experimental Methodology

Our experimental methodology is based on that of Falcón-Cardona and Coello Coello [6]. Following [6], we use 7 problems (DTLZ1 through DTLZ7) from the Deb-Thiele-Laummans-Zitzler (DTLZ) [8] suite and 9 problems (WFG1 through WFG9) from the Walking-Fish-Group (WFG) [9] suite. For each problem, we use four values for the number of objectives (3, 5, 7, and 10), for a total of 64 problem instances. For each instance, we run each of the algorithms under evaluation for 30 trials. We use the parameter settings of Falcón-Cardona and Coello Coello [6], summarized in Table I.

In our comparison, performance is assessed with the hypervolume (HV) indicator. We use the HV implementation of [15], available in [16]. Computing the HV requires that a reference vector be supplied by the user. We use the set of reference vectors that were used in [6], which we repeat in Table II for convenience. Occasionally, particularly for DTLZ1 and DTLZ3, the reference vector dominates all the solutions

TABLE I
PARAMETER SETTINGS.

Parameter	Value	
N, M	120/126/84/220	for 3/5/7/10 objectives
q	0.1	
ξ	0.5	
G_{max}	416/396/595/227	for 3/5/7/10 objectives
α	0.5	
ϵ	0.001	
h	14/5/7/19	for 3/5/7/10 objectives

TABLE II
REFERENCE VECTORS USED WITH HYPERVOLUME INDICATOR.

problem	vector	problem	vector
DTLZ1	$(1, 1, \ldots)$	DTLZ5	$(4, 4, \ldots)$
DTLZ2	$(2, 2, \ldots)$	DTLZ6	$(11, 11, \ldots)$
DTLZ3	$(7, 7, \ldots)$	DTLZ7	$(1, 1, \ldots, 21)$
DTLZ4	$(2, 2, \ldots)$	WFG (all)	$(3, 5, 7, \ldots, 2m + 1)$

returned by the algorithm under evaluation. Following the approach of [6], HV is taken as zero in such cases.

We evaluate and compare three algorithms:

1) The original iMOACO$_{\mathbb{R}}$ algorithm [6].
2) The iMOACO$_{\mathbb{R}}$-**R** algorithm which applies recombination with a fixed probability P_r. We use $P_r = 0.2$ based on previous work [7] which found that setting to perform well.
3) Our proposed iMOACO$_{\mathbb{R}}$-**AR** algorithm which adapts the frequency of applying recombination based on the recent performance of the recombination operators.

VI. RESULTS

We run each of the three algorithms (iMOACO$_{\mathbb{R}}$, iMOACO$_{\mathbb{R}}$-**R**, iMOACO$_{\mathbb{R}}$-**AR**) for 30 trials on each of the 64 problem instances described in Section V, and compute the value of the hypervolume (HV) indicator in each case. Table III reports the mean and standard deviation of HV for each of the algorithms for each problem instance. In each row, the best value of mean HV is shown in boldface. The last row shows the average rank for each algorithm. For each algorithm a, the rank of a is first obtained individually for each problem instance; in the case of ties, each of the tied algorithms is given the average of the spanned ranks. The individual ranks are then averaged across all 64 instances to obtain the average rank for each algorithm. The lower the rank, the better. The table indicates that the best rank is obtained by iMOACO$_{\mathbb{R}}$-**AR** with a rank of 1.6, followed closely by iMOACO$_{\mathbb{R}}$-**R** with a rank of 1.7, while iMOACO$_{\mathbb{R}}$ has a rank of 2.7.

The results indicate the following:

- iMOACO$_{\mathbb{R}}$-**AR** performs slightly better than iMOACO$_{\mathbb{R}}$-**R**, with 34 wins, 29 losses, and 3 ties.
- iMOACO$_{\mathbb{R}}$-**AR** also performs better than iMOACO$_{\mathbb{R}}$, with 53 wins, 9 losses, and 2 ties.
- iMOACO$_{\mathbb{R}}$-**R** performs better than iMOACO$_{\mathbb{R}}$, with 56 wins, 7 losses, and 1 tie.

TABLE IV
RESULTS OF WILCOXON SIGNED-RANK STATISTICAL SIGNIFICANCE TESTS.

Comparison	W	#	z	p	sig.?
iMOACO$_{\mathbb{R}}$-**AR** vs iMOACO$_{\mathbb{R}}$-**R**	799	62	-1.245	0.215	no
iMOACO$_{\mathbb{R}}$-**AR** vs iMOACO$_{\mathbb{R}}$	249	62	-5.101	$<$ 0.000	yes
iMOACO$_{\mathbb{R}}$-**R** vs iMOACO$_{\mathbb{R}}$	232	63	-5.313	$<$ 0.000	yes

Table IV shows the results of applying Wilcoxon signed-rank tests comparing each pairing of the three algorithms. For each comparison, the table shows the computed value of the W statistic, the number of samples (after excluding ties), and the corresponding values of z and p. Conventionally, p-values less than or equal to 0.05 are considered statistically significant. As indicated in the last column of the table, the improvement of iMOACO$_{\mathbb{R}}$-**AR** over iMOACO$_{\mathbb{R}}$-**R** is not statistically significant. Of course, iMOACO$_{\mathbb{R}}$-**AR** still has the advantage of not requiring a P_r parameter. On the other hand, both iMOACO$_{\mathbb{R}}$-**AR** and iMOACO$_{\mathbb{R}}$-**R** have a significant difference over iMOACO$_{\mathbb{R}}$.

The reasons iMOACO$_{\mathbb{R}}$-**AR** performs better than iMOACO$_{\mathbb{R}}$-**R** are likely because of its adaptation mechanism. iMOACO$_{\mathbb{R}}$-**R** uses a fixed probability of recombination P_r that may not be the best setting for every problem instance, whereas iMOACO$_{\mathbb{R}}$-**AR** determines the probability of recombination automatically during the algorithm's execution. Furthermore, iMOACO$_{\mathbb{R}}$-**AR** allows the probability of applying recombination to increase and decrease over the course of the computation, to be better coupled to the characteristics of different regions of the search space, whereas in iMOACO$_{\mathbb{R}}$-**R**, P_r is fixed throughout the computation.

However, one might ask the question: why are there instances, such as DTLZ5 (for 5, 7, and 10 dimensions), where the non-adaptive iMOACO$_{\mathbb{R}}$-**R** performs better than iMOACO$_{\mathbb{R}}$-**AR**? We would postulate that these are problem instances for which $P_r = 0.2$ happens to be a near-ideal setting, and remains near-ideal throughout the computation. The adaptive iMOACO$_{\mathbb{R}}$-**AR** will take time in order to discover a good setting, because the probability of recombination starts off at a very small value (as discussed earlier).

It should be emphasized that we are evaluating iMOACO$_{\mathbb{R}}$-**R** using a setting of P_r that was determined in previous work [7] through experimentation with different possible values of P_r, and is therefore already known to be a good general setting. In contrast, iMOACO$_{\mathbb{R}}$-**AR** is not given any external insight and has to discover a good value of P_r on its own.

The fact that iMOACO$_{\mathbb{R}}$-**AR** performs better than the original iMOACO$_{\mathbb{R}}$ algorithm is not surprising, since it was established in previous work [7] that the non-adaptive iMOACO$_{\mathbb{R}}$-**R** performs better than iMOACO$_{\mathbb{R}}$.

Figure 2 presents a comparison of Pareto fronts produced by the three algorithms. For three objective functions, it is not completely clear why iMOACO$_{\mathbb{R}}$-**AR** obtains the best hyper-

TABLE III
THE MEAN AND STANDARD DEVIATION OF HV FOR EACH OF THE THREE ALGORITHMS UNDER EVALUATION.

instance	dim	mean			std. dev.		
		iMOACO$_{\mathbb{R}}$-**AR**	iMOACO$_{\mathbb{R}}$-**R**	iMOACO$_{\mathbb{R}}$	iMOACO$_{\mathbb{R}}$-**AR**	iMOACO$_{\mathbb{R}}$-**R**	iMOACO$_{\mathbb{R}}$
DTLZ1	3	1.309731e-01	**3.405737e-01**	0.000000e+00	2.043084e-01	3.065881e-01	0.000000e+00
	5	4.024716e-01	**5.178174e-01**	0.000000e+00	3.282377e-01	3.367047e-01	0.000000e+00
	7	1.218345e-01	**3.239936e-01**	0.000000e+00	2.514043e-01	4.068167e-01	0.000000e+00
	10	7.126941e-01	**7.667274e-01**	0.000000e+00	3.442033e-01	3.015299e-01	0.000000e+00
DTLZ2	3	**7.421963e+00**	7.421954e+00	7.420235e+00	1.135140e-04	1.749650e-04	3.140000e-04
	5	**3.166738e+01**	3.166680e+01	3.164923e+01	7.674060e-04	6.929420e-04	1.955981e-03
	7	1.273375e+02	**1.276548e+02**	1.272041e+02	2.022452e+00	1.761318e-01	1.376955e+00
	10	1.019795e+03	**1.021218e+03**	1.013835e+03	1.149012e+01	3.755776e+00	3.038910e+01
DTLZ3	3	0.000000e+00	**2.888276e+01**	0.000000e+00	0.000000e+00	7.488256e+01	0.000000e+00
	5	1.647957e+00	**2.918011e+02**	0.000000e+00	4.972835e+00	1.169783e+03	0.000000e+00
	7	**0.000000e+00**	**0.000000e+00**	**0.000000e+00**	0.000000e+00	0.000000e+00	0.000000e+00
	10	**4.509804e+07**	4.349740e+07	0.000000e+00	8.330747e+07	7.604062e+07	0.000000e+00
DTLZ4	3	7.420607e+00	**7.420882e+00**	7.418887e+00	7.651660e-04	8.386120e-04	9.160000e-04
	5	**3.166336e+01**	3.166120e+01	3.163343e+01	2.049169e-03	2.905367e-03	5.217947e-03
	7	**1.277362e+02**	1.276846e+02	1.265156e+02	4.432059e-02	1.786903e-01	4.154677e+00
	10	1.015039e+03	**1.021865e+03**	1.003038e+03	2.039062e+01	2.153971e+00	4.235631e+01
DTLZ5	3	5.983700e+01	**5.983868e+01**	5.983854e+01	9.896603e-03	7.714740e-03	7.923007e-03
	5	9.342031e+02	9.338535e+02	**9.374394e+02**	2.049481e+00	1.124213e+00	9.097390e-01
	7	1.415837e+04	1.424640e+04	**1.438291e+04**	2.895692e+02	3.179964e+02	1.064769e+02
	10	9.125468e+05	9.285390e+05	**9.362118e+05**	1.325129e+04	5.820858e+03	6.302014e+03
DTLZ6	3	**1.318857e+03**	1.318852e+03	1.316253e+03	8.089254e-03	6.675449e-03	1.324603e+00
	5	1.588226e+05	**1.589105e+05**	1.567570e+05	2.620255e+02	5.152005e+01	1.004394e+03
	7	**1.869788e+07**	1.858547e+07	1.734328e+07	2.840866e+05	2.195254e+05	1.914507e+06
	10	2.504094e+10	**2.529153e+10**	2.386460e+10	2.911453e+08	1.558542e+08	1.446229e+09
DTLZ7	3	**1.635495e+01**	1.632627e+01	1.624761e+01	7.813830e-04	5.887169e-02	6.003250e-02
	5	1.293432e+01	**1.295728e+01**	1.256260e+01	7.713103e-02	1.173998e-01	1.093206e-01
	7	7.994842e+00	**8.383287e+00**	8.238742e+00	4.354401e-01	2.854881e-01	1.949213e-01
	10	2.650328e+00	**2.698024e+00**	1.463875e+00	2.318165e-01	1.935057e-01	7.508288e-01
WFG1	3	**5.672217e+01**	5.049341e+01	4.416989e+01	1.681969e+00	9.528380e-01	6.970353e-01
	5	**4.800143e+03**	4.536323e+03	3.923216e+03	1.957881e+02	6.561412e+01	9.186329e+01
	7	**1.012960e+06**	8.453257e+05	6.693291e+05	4.657313e+04	7.355926e+03	3.308836e+04
	10	**4.655908e+09**	4.267833e+09	3.968796e+09	2.520808e+07	1.379010e+07	2.069459e+07
WFG2	3	**9.933827e+01**	9.811442e+01	9.744023e+01	7.625626e-01	8.470477e-01	5.512647e-01
	5	1.022450e+04	**1.024813e+04**	9.706916e+03	4.409717e+01	3.763713e+01	9.077778e+01
	7	1.888481e+06	**1.920465e+06**	1.693878e+06	2.044528e+04	1.827917e+04	2.784520e+04
	10	**1.257257e+10**	1.116778e+10	9.467046e+09	1.019488e+08	1.354532e+08	1.332691e+08
WFG3	3	**7.466014e+01**	7.417588e+01	7.245173e+01	2.356532e-01	3.109563e-01	2.456975e-01
	5	4.957151e+03	5.126415e+03	**5.390769e+03**	4.152642e+02	1.383951e+02	2.499760e+02
	7	6.557288e+05	7.474997e+05	**7.839258e+05**	1.106508e+05	7.243166e+04	1.962740e+04
	10	4.570736e+09	4.574113e+09	**4.799790e+09**	1.590708e+08	1.344666e+08	2.360311e+08
WFG4	3	**7.571198e+01**	7.523355e+01	7.066606e+01	2.601592e-01	3.348561e-01	3.457368e-01
	5	**8.572311e+03**	8.459619e+03	7.614682e+03	1.123784e+02	1.377473e+02	1.558473e+02
	7	1.456661e+06	**1.495149e+06**	1.241569e+06	8.503477e+04	5.727180e+04	5.108251e+04
	10	**1.001431e+10**	8.829287e+09	7.219131e+09	9.228356e+08	5.266028e+08	3.059377e+08
WFG5	3	**7.103873e+01**	7.084212e+01	6.830671e+01	3.614418e-01	4.554669e-01	7.109270e-01
	5	**6.993621e+03**	6.903642e+03	4.786266e+03	2.783963e+02	3.414359e+02	1.682254e+02
	7	1.153791e+06	**1.166619e+06**	6.938271e+05	1.219488e+05	6.249238e+04	3.439495e+04
	10	**7.798253e+09**	5.487968e+09	4.326382e+09	4.842156e+08	2.896484e+08	2.108786e+08
WFG6	3	7.343451e+01	7.403118e+01	**7.414183e+01**	7.806236e-01	4.955407e-01	3.567494e-01
	5	**8.412675e+03**	8.073424e+03	6.673660e+03	1.838619e+02	2.865328e+02	3.394313e+02
	7	1.558772e+06	**1.586780e+06**	8.465877e+05	4.321214e+04	7.209466e+04	7.416890e+04
	10	**1.018512e+10**	7.029743e+09	4.796231e+09	7.476273e+08	6.041572e+08	2.441963e+08
WFG7	3	**7.708252e+01**	7.571637e+01	7.522232e+01	1.262511e-01	3.778515e-01	2.566370e-01
	5	**8.896706e+03**	8.658600e+03	7.214164e+03	3.554640e+01	8.293910e+01	2.731626e+02
	7	**1.727071e+06**	1.694156e+06	1.085651e+06	2.311153e+04	3.419381e+04	5.986574e+04
	10	**1.145889e+10**	8.953337e+09	6.961461e+09	3.337603e+08	2.475337e+08	2.715088e+08
WFG8	3	**6.779437e+01**	6.650534e+01	6.541337e+01	3.274858e-01	3.378578e-01	2.995898e-01
	5	**5.994972e+03**	5.775872e+03	5.158155e+03	2.471690e+02	1.900633e+02	2.691419e+02
	7	**1.089499e+06**	1.061901e+06	7.796808e+05	1.134469e+05	7.357747e+04	6.838321e+04
	10	**8.326035e+09**	7.043867e+09	5.180671e+09	3.472185e+08	3.376537e+08	4.100537e+08
WFG9	3	**6.609922e+01**	6.603353e+01	6.594158e+01	1.695498e-01	2.174347e-01	1.833654e-01
	5	6.180476e+03	**6.385138e+03**	5.851314e+03	5.625705e+02	2.152490e+02	4.018221e+02
	7	8.483316e+05	**8.942911e+05**	7.113797e+05	1.286020e+05	1.701840e+05	1.049130e+05
	10	**5.688024e+09**	5.206364e+09	4.161832e+09	9.999015e+08	7.231623e+08	3.453454e+08
rank (avg)		**1.61**	1.66	2.73			

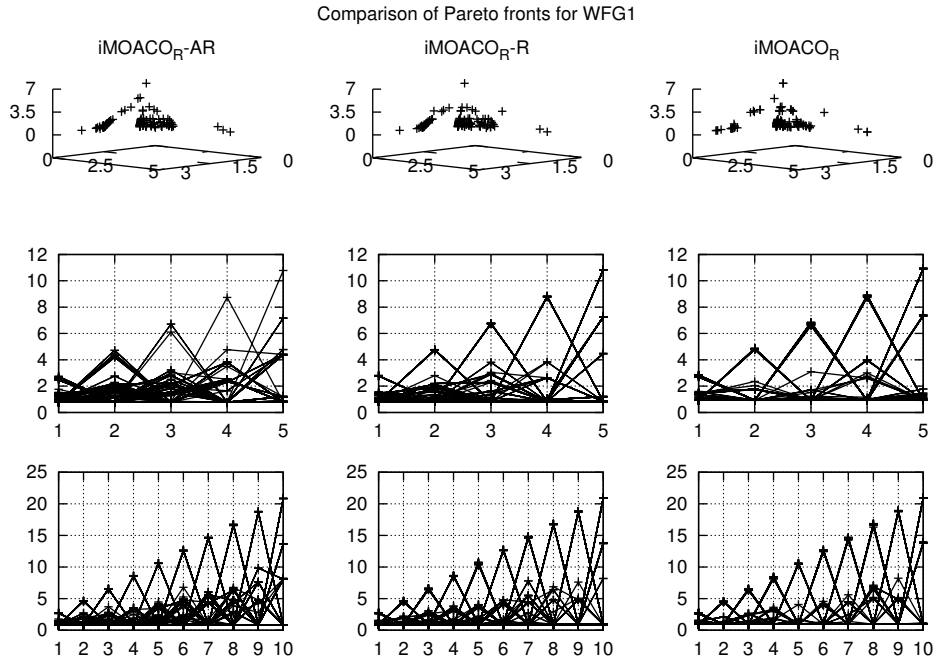

Fig. 2. Pareto fronts for WFG1 test problem for 3, 5 and 10 objective functions. Each column is related to one of the compared algorithms. The Pareto fronts correspond to the hypervolume median.

volume value. However, this is related to a better coverage of the Pareto front's knee and better boundary solutions. This fact is more clear for 5 and 10 objective functions, where iMOACO$_\mathbb{R}$-**AR** obtains the better hypervolume value as well. This observation is more evident when iMOACO$_\mathbb{R}$-**AR** is compared to iMOACO$_\mathbb{R}$. Hence, based on these results and the ones of the remaining problems, we claim that iMOACO$_\mathbb{R}$-**AR** produces better point distributions than iMOACO$_\mathbb{R}$-**R** and iMOACO$_\mathbb{R}$.

VII. Concluding Remarks

Multi-Objective Ant Colony Optimization (MOACO) for continuous search spaces is a relatively new research area since most of the MOACO effort has been focused on combinatorial problems. However, two years ago, a new algorithm called iMOACO$_\mathbb{R}$ was introduced for solving continuous MOPs in low- and high-dimensional objective spaces. One important drawback of iMOACO$_\mathbb{R}$ is related to the way that ants create new solutions. Hence, a previous work [7], denoted as iMOACO$_\mathbb{R}$-**R**, indicated that the iMOACO$_\mathbb{R}$ algorithm benefits from the inclusion of a mechanism which applies recombination with some probability. In this work, we have presented an adaptation mechanism that adjusts the probability of applying recombination automatically based on the quality of solutions constructed by recombination. The quality of such solutions is judged indirectly based on how long the solutions survive in the archive before being displaced, and their relative rank in the archive. Our experimental results suggest that this approach, called iMOACO$_\mathbb{R}$-**AR** is effective in adapting the probability of recombination without external input from the user. Moreover, the comparison based on the hypervolume indicator shows that iMOACO$_\mathbb{R}$-**AR** outperforms iMOACO$_\mathbb{R}$-**R** and iMOACO$_\mathbb{R}$ in most of the adopted test problems, making it, to the authors' best knowledge, the best MOACO for continuous search spaces in the literature. However, there is still room for improvement. As part of our future work, we are interested in studying new probability density functions to generate better solutions in multifrontal MOPs and an adaptive method for the parameters q and ξ is required.

Acknowledgements

The partial support of a research grant from the Brandon University Research Council (BURC) is gratefully acknowledged. The authors wish to thank C.M. Fonseca, L. Paquete and M. López-Ibáñez for making their tool for computing the hypervolume indicator publicly available.

References

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. New York, NY, USA: Oxford University Press, Inc., 1999.

[2] M. Dorigo and T. Stützle, *Ant Colony Optimization*. MIT Press, 2004.

[3] K. Miettinen, *Nonlinear Multiobjective Optimization*. Kluwer Academic Publisher, 1999.

[4] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed. New York: Springer, September 2007, iSBN 978-0-387-33254-3.

[5] T. Liao, K. Socha, M. Montes de Oca, T. Stützle, and M. Dorigo, "Ant colony optimization for mixed-variable optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 18, pp. 503–518, 2014.

[6] J. G. Falcón-Cardona and C. A. Coello Coello, "A new indicator-based many-objective ant colony optimizer for continuous search spaces," *Swarm Intelligence*, vol. 11, pp. 71–100, 2017.

[7] A. M. Abdelbar and K. M. Salama, "Solution recombination in an indicator-based many-objective ant colony oprimizer for continuous search spaces," in *Proceedings IEEE Swarm Intelligence Symposium*, 2017.

[8] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," in *Proceedings CEC-2002*, vol. 1, 2002, pp. 825–830.

[9] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Transactions on Evolutionary Computation*, vol. 10, pp. 477–506, 2006.

[10] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, April 2003.

[11] D. Brockhoff, T. Wagner, and H. Trautmann, "On the properties of the R2 indicator," in *2012 Genetic and Evolutionary Computation Conference (GECCO'2012)*. Philadelphia, USA: ACM Press, July 2012, pp. 465–472, iSBN: 978-1-4503-1177-9.

[12] R. Hernández Gómez and C. A. Coello Coello, "Improved metaheuristic based on the R2 indicator for many-objective optimization," in *2015 Genetic and Evolutionary Computation Conference (GECCO 2015)*. Madrid, Spain: ACM Press, July 11-15 2015, pp. 679–686, iSBN 978-1-4503-3472-3.

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, April 2002.

[14] A. Kalinli and F. Sarikoc, "A parallel ant colony optimization algorithm based on crossover operation," in *Advances in Metaheuristics for Hard Optimization*. Berlin: Springer, 2008, pp. 87–110.

[15] C. M. Foncesca, L. Paquete, and M. López-Ibáñez, "An improved dimension-sweep algorithm for the hypervolume indicator," in *Proceedings CEC-2006*, 2016, pp. 1157–1163.

[16] C. M. Fonsesca, M. López-Ibáñez, L. Paquete, and A. P. Guerreiro, "Computation of the hypervolume indicator," http://iridia.ulb.ac.be/~manuel/hypervolume, accessed: May 2017.