A Study of Objective Weights in Multi-Objective Ant Colony Optimization

Andrew Runka
Carleton University
Ottawa, Ontario, Canada
arunka@connect.carleton.ca

Tony White
Carleton University
Ottawa, Ontario, Canada
arpwhite@scs.carleton.ca

ABSTRACT

This paper examines the use of objective-biasing weights in Multi-Objective Ant Colony Optimization (MOACO). Past literature pertaining to objective weights in MOACO is examined. A new formalism is proposed to consolidate descriptions of various weight setting strategies. Experimentation is performed to analyze consequences of various objective weight design choices in terms of their effect on solution quality and fitness space coverage.

1. INTRODUCTION

In the endeavour to create artificial intelligence techniques that are applicable to real world problems, the need to tackle multi-objective domains becomes emphasized. The additional complexity of multiple objectives over typical single-objective problems makes this area particularly enticing for research using heuristic optimization techniques. A number of multi-objective extensions to known single-objective heuristic optimization techniques such as Ant Colony Optimization (ACO), Genetic Algorithms, and Particle Swarm Optimization have been developed and improvements to these continue to surface over time [2, 5, 13, 16, 21, 22, 23].

This paper focusses on the application of the ACO metaheuristic to problems in the multi-objective domain. The relevant body of literature has expanded rapidly since the inception of this topic. As such, a large number of comparisons, reviews, and taxonomies have been presented. Proceeding from these, this paper explores the design decisions and consequences related to various strategies for setting objective-biasing weights. Strategies are compared based on their relative effects on solution quality, exploration, and exploitation, given a common algorithmic base.

The contributions of this work are:

- A survey of existing MOACO taxonomies, and various objective weight setting strategies found in the literature,

- An expansion of the formal description of MOACO algorithms, and

- An analysis of the impacts of weight setting design choices.

The remainder of the paper is organized as follows. The next section presents background information on related topics such as multi-objective optimization, the test problems to be used, and the Ant Colony Optimization metaheuristic. Following this, Section 3 presents a survey of literature relevant to this study. The implementation details and other setup related information are described in Section 4. Section 5 contains the findings of this paper, and Section 6 presents the concluding remarks.

2. BACKGROUND

This section presents background information for the concepts required of the rest of the paper. A reader well-versed in these topics may skip the corresponding sections without detriment to their understanding of this paper.

2.1 Multi-Objective Combinatorial Optimization

When dealing with multiple (and often conflicting) objectives, it is typically impractical to name a single best solution. To this end, the set of desirable solutions along a trade-off surface, known as the *pareto front*, are sought for optimality. Using heuristic optimization techniques, an approximation to the Pareto front is developed and returned as the best results of a given run. This is known as the set of non-dominated solutions. A solution a dominates a solution b (in terms of a maximization problem) if the following condition is true:

$$\forall g \in G : f_g(a) \geq f_g(b) \ and \ \exists g^* \in G : f_{g^*}(a) > f_{g^*}(b) \quad (1)$$

where G is the set of all objectives, and $f_g(a)$ is the fitness of solution a with respect to objective g. That is, a solution is dominated if there exists another solution that is not worse in any objective, and better in at least one objective. A solution is said to be non-dominated if it is not dominated by any other candidate solution. The set of all discovered non-dominated solutions is used as an estimate of the Pareto front.

As a result of having a set of best solutions to a given problem, it is not readily apparent which technique performs better in a direct comparison. Traditional means of statistical comparison are ineffective in this multi-dimensional domain. To address this issue, this paper uses the following multi-objective comparison techniques.

2.1.1 *Coverage*

The most straight-forward comparison metric used is the coverage of two sets. This is the calculation of the ratio of solutions from one set which are weakly dominated by solutions from a second set. Given two sets of solutions X' and X'', the coverage is calculated as follows.

$$C(X', X'') = \frac{|\{a'' \in X'', \exists a' \in X' : a' \succeq a''\}|}{|X''|} \quad (2)$$

This reduces the multi-dimensional comparison to a value in the range [0,1] where C(X′,X″)=1 means that all points in X" are dominated by points in X′. It should be noted, however, that the coverage calculation is not symmetric. That is, C(X′,X″) $\neq$ 1-C(X″,X′). So for comparison purposes both C(X′,X″) and C(X″,X′) are calculated and presented.

2.1.2 Hypervolume

The hypervolume metric was proposed by Zitzler et al. [25] as a means of evaluating a Pareto-front's quality with respect to the spread of solutions (as opposed to the density of good solutions, potentially in only a small area). The hypervolume of a set of solutions is calculated as the volume of a union of hyper-rectangles defined between each solution (n-dimensional point in fitness space) and the origin[1]. The hypervolume only requires a single set of solutions for calculation; however, given two sets of solutions, the dominated volumes of each set over one another can be calculated. This provides a straightforward means of comparison of the comprehensiveness of each solution set.

2.1.3 Univariate analysis

The third technique applied to the comparison and analysis of Pareto fronts is referred to here as univariate analysis. This technique was proposed by Fonseca and Fleming [12] and is further explained in [18]. Univariate analysis is a multi-step approach aimed at reducing the dimensionality of the solution sets so that traditional statistical operators can then be applied. The first step, given a solution set, is to determine the attainment surface. The attainment surface is defined as the set of hyperplanes along the dimension of each objective that connect the solution set in a way that makes minimal assumptions about the the quality of solutions in between points of the set. Once the attainment surface is defined for all pareto fronts to be compared, then a number of sample vectors are drawn from the origin through the attainment surfaces. The points of intersection along the each sample vector define a 1-dimensional (univariate) distribution of the various surfaces being considered. Thus, given several fronts from each algorithm to be compared, traditional statistical comparisons can be applied. The implementation used in this paper[2] compares the algorithms using a Mann-Whitney-Wilcoxon (MWW) rank-sum test with a 95% confidence interval.

Both hypervolume and univariate metrics are based on calculating the attainment surface, the difference being that the former calculates the raw difference between two surfaces, while the latter performs a statistical analysis on *samples* of the attainment surface.

2.2 Multi-Objective Knapsack Problem

As a preliminary, the *single-objective* knapsack problem can be described as follows: Given a set of objects, each with a given weight and profit value, select the subset of those objects which maximizes the total profit without exceeding the weight capacity of the knapsack.

Using that description, the *multi-objective* knapsack problem (MOKP) can be described as using a single index vector to simultaneously solve an arbitrary number of single-objective knapsack problems. Formally:

$$Maximize: \quad f^h = \sum_{i=1}^{n} p_i^h x_i \quad \forall h \in 1...H \quad (3)$$
$$Subject\ to: \quad \sum_{i=1}^{n} w_i^h x_i \leq c^h \quad \forall h \in 1...H \quad (4)$$
$$x_i \in 0,1 \quad i = 1,...,n \quad (5)$$

where n is the number of objects in the given instance, H is the number of objectives, p_i^h is the profit of the i^{th} object with respect to objective h, w_i^h is the weight of the i^{th} object with respect to objective h, c^h is the weight capacity of knapsack h, and x is a solution vector comprised of $\{0,1\}$ such that $x_i = 0$ means the the i^{th} object is not included in the solution subset and $x_i = 1$ means that it is.

An individual knapsack exists for each objective, and each knapsack is associated with its own weight capacity. There exists an independent set of objects for each objective, all with their own weights and profit values, but with shared indexes. Thus, a value of 1 in an element of the x vector adds one object from each objective to its corresponding knapsack.

The MOKP instances used in this paper were first presented in [26], and are available online[3].

2.3 Single-Objective Ant Colony Optimization

Ant Colony Optimization (ACO) is a metaheuristic modelled on the natural optimization behaviour of real ants. In reality, a population of ants cooperate by use of pheromone trails to find optimal paths between a nest and a food source. The concept of pheromone was borrowed for ACO to act as a means of balancing between exploration and exploitation in a combinatorial optimization search space. The ACO metaheuristic can be broken down into three main phases: *generate solutions*, *daemon actions*, and *update pheromone*. Several instantiations of the ACO metaheuristic exist. This paper addresses an extension to the simplest such instantiation, known as the Ant System (AS), thus details described here pertain to the AS implementation where pertinent. For an in-depth description of AS and its extensions see [11].

The first phase of the ACO metaheuristic encompasses the construction of solutions to the given problem. Solutions are in the form of paths through the problem graph. At each step during construction each ant adds one vertex to its path using roulette wheel selection. The probability of moving from the current vertex i, to an adjacent vertex j in the neighbourhood of i is calculated as follows:

$$p_{ij} = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{h \in N_{i,k}} [\tau_{ih}]^\alpha \cdot [\eta_{ih}]^\beta} \forall j \in N_{ik} \quad (6)$$

where N_{ik} is the neighbourhood of vertex i given the current

[1] Hypervolume implementation available online at: http://www.tik.ee.ethz.ch/sop/download/supplementary/

[2] Univariate analysis implementation available online at: http://dbkgroup.org/knowles/multi/

[3] The MOKP instances are available online: http://www.tik.ee.ethz.ch/~zitzler/testdata.html

state of the solution constructed by ant k, τ_{ij} is the amount of pheromone on $edge_{ij}$, and η_{ij} is the desirability of $edge_{ij}$ which is the *a priori* information known about the problem. The exponents α and β are parameters which control the degree of exploration versus exploitation via weighting the influence of τ and η on the final probability respectively.

The second phase of the ACO metaheuristic, daemon actions, is an optional phase. This is used to perform any post-processing steps, such as local search, to the solutions generated in the previous phase prior to evaluation.

Once each ant in the population has constructed a solution (and optionally performed daemon actions), the third phase, known as the pheromone update, takes place. The pheromone update can be separated into two steps: evaporation and deposit. In the evaporation step, the decayed amount of pheromone is calculated as follows:

$$\tau_{ij} = (1-\rho)\tau_{ij} \tag{7}$$

where τ_{ij} is the amount of pheromone on edge$_{ij}$, and ρ is a parameter in the range [0,1] that controls the rate of evaporation. This equation is applied globally to all edges in the graph.

The second step, deposit, is calculated for each ant over the path it took through the graph, and for maximization problems typically takes the form:

$$\Delta\tau_{ij,k} = \begin{cases} 1 - \frac{1}{f(\psi_k)}, & \text{if ant k traverses edge}_{ij} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where $f(\psi_k)$ is the fitness of the kth ant's solution.

Through the iterative application of Equations 6, 7, and 8, the amount of pheromone will accumulate on edges which appear to be parts of good solutions to the problem thus attracting future ants towards those edges.

3. MULTI-OBJECTIVE ANT COLONY OPTIMIZATION

In the past decade or so, a large number of multi-objective applications of the ACO metaheuristic have been presented. Most of which were presented as unique innovations without relation to one another. Following from this trend, a number of taxonomies have been proposed in order to categorize and unify the various approaches. This section first covers a number of these taxonomies in a bid to describe the fundamental ideas of MOACO. Following this, several of the individual approaches pertinent to this study are examined. It should be noted that the treatment of these approaches is focussed primarily on the techniques employed for aggregation of information relating to multiple objectives. The interested reader looking for a more detailed study of the past MOACO approaches is encouraged to review [5], [13], and [20]. Equations that are reproduced in this section may have been modified to maintain consistency of presentation, but their meaning remains unaltered. All reproductions of edge selection formulas are subject to the typical neighbourhood constraint, although it is not displayed in Equations 10 or 20 for brevity.

The most current taxonomy as of writing this paper is presented by Angus and Woodward [5]. The proposed taxonomy is designed to remain close to the traditional ACO metaheuristic while addressing the many possible ways of extending the metaheuristic to the multi-objective domain. It contains five categories of MOACO algorithm components that can be loosely mapped to stages of an ACO algorithm's execution. An initialization phase of such a taxonomic algorithm would consist of the definition of single or multiple pheromone matrices. Not mentioned in the taxonomy is the analogous decision for the heuristic matrices. Following this initialization, each ant constructs a solution. This category is described in greater detail below. Once an MOACO algorithm has constructed a population of solutions they are evaluated. This category distinguishes between algorithms that use Pareto dominance in evaluation and those that do not (e.g., solutions are evaluated for a single objective, or using a technique such as weighted sum). Once solutions are evaluated, the taxonomic algorithm undergoes a *daemon action* stage where it is decided how to maintain the population of non-dominated solutions. Whether solutions are kept aside as a record of the best-so-far in a 'Hall of Fame' style, are kept more closely tied to pheromone update procedures, or only kept long enough to update the pheromone matrices, is distinguished in this category of the taxonomy. Following daemon actions, both the ACO metaheuristic and the proposed taxonomy dictate that the pheromone matrix(/matrices) is to be updated. If multiple pheromone matrices are being used then the taxonomy distinguishes between pheromone updates in a single matrix or in multiple (or all) matrices at once. While this taxonomy provides a solid basis for exploring the variations of MOACO algorithms, it does not contributed to an experimental analysis of the categories considered.

The category of this taxonomy that is of particular relevance to the study at hand is the *solution construction* category. This largely describes how various MOACO algorithms handle the aggregation of information pertaining to multiple objectives using the edge selection formula. In this category the algorithms are one of three types: Targeted, Dynamic, or Fixed. Targeted refers to algorithms that use specific objectives to construct solutions; in other words, the objectives are never aggregated. Fixed means that the ratios or weighting used for aggregation are set *a priori* and remain constant throughout execution. Dynamic is a general term referring to the fact that the specific bias of construction is capable of changing during execution; for example, by assignment of different objective weights per ant, or by varying a single set of global weights. This is a fairly broad categorization encompassing the majority of MOACO algorithms considered. Looking deeper into this category could yield more classifications between implementations with regards to their use of weight setting strategies. The taxonomy does go a step further and presents a generalized form of the ACO decision formula tailored to dynamically constructed multi-objective optimization, reproduced below.

$$p_{ij,k} = \begin{cases} \dfrac{\prod_{a=1}^{A}[\tau_{ij,a}]^{\alpha_a} \cdot \prod_{b=1}^{B}[\eta_{ij,b}]^{\beta_b}}{\sum_{h\in N_{i,k}} \prod_{a=1}^{A}[\tau_{ih,a}]^{\alpha_a} \cdot \prod_{b=1}^{B}[\eta_{ih,b}]^{\beta_b}}, & \text{if } j \in N_{i,k} \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where $p_{ij,k}$ is the probability of moving from node i to node j for ant k, A and B are the number of pheromone matrices and heuristic matrices respectively, $\tau_{ij,a}$ is the pheromone from matrix a on edge$_{ij}$, $\eta_{ij,b}$ is the b^{th} heuristic desirability measure associated with edge$_{ij}$, α_a is the weighting associated with pheromone matrix a, β_b is the weighting associated with heuristic measure b, and $N_{i,k}$ is the accessible neighbourhood for ant k from node i. The values for A and B are commonly set to either 1 or the number of objectives. It should be noted that while the above equation aggregates information as a product of objective values, this is only one potential mechanism. This matter is discussed in more detail at the end of this section.

García-Martínez et al.[13] propose a two-dimensional taxonomy of MOACO algorithms based on the number of pheromone matrices and heuristic matrices in the design of each. In [13] a comprehensive listing of MOACO algorithms that precede it are provided. A reader interested in studying the variety of MOACO algorithms is encouraged to read this review. The considered MOACO algorithms are compared based on their performance at solving instances of the bi-objective TSP. It is concluded that the proposed taxonomy does not accurately describe the observable differences in the solution qualities of each MOACO algorithm. Additional observations includes the favourable quality of MOACO results as compared to the state-of-the-art Multi-Objective Genetic Algorithm (MOGA) results for the problems considered. Also, while the results between MOACO algorithms are presented to be of similar quality to each other, three algorithms are mentioned as being the most promising: P-ACO, MONACO, and the so-called BicriterionMC (the update by region method of the multi-colony approach proposed for the Bicriterion Ant Algorithm), each are described later in this section. Interestingly, these three approaches are all classified as using 'dynamic' solution construction in the previously mentioned taxonomy by Angus and Woodward.

López-Ibáñez et al.[20, 19] analyze several components of MOACO algorithms in terms of the effect the design decisions have on the quality of solutions found for the bi-objective TSP and bi-objective Quadratic Assignment Problem (QAP). López-Ibáñez et al. examine Local Search, Pheromone Information, Weight Setting Strategies, Pheromone Deposit, and Multiple Colonies as the categorical distinctions. Of particular relevance to this paper is their treatment of Weight Setting Strategies used to aggregate information from multiple pheromone matrices. Two such strategies are considered: either all ants use the same weights which shift, or every ant is given its own set of weights (linearly distributed over the two objectives) which remain static. The resulting conclusion is that a single set of shifting weights obtains superior results when compared to using a static set of weights for each ant. These two weight setting strategies are a non-exhaustive sample of the set of all possible strategies. A similar comparison involving additional strategies is the focus of later sections of this paper.

Alaya et al.[1] list three main categorical distinctions of ACO applications to multi-objective problems: the number of pheromone matrices, the number of heuristic measures, and the selection of solutions to reward. According to this taxonomy, there are two approaches to extending the traditional single pheromone matrix to multi-objective domains. One is to aggregate the information for all objectives into a single pheromone matrix using a technique such as weighted sum. The second approach is to maintain multiple pheromone matrices, one for each objective. Again, this takes two forms. In one situation ants may be assigned to a specific pheromone matrix, as is often the case when using multiple ant colonies. This then necessitates an additional cooperation mechanism for the colonies to share their objective-based knowledge. Alternatively, one might maintain an independent pheromone matrix per objective, then aggregate the knowledge from the separate matrices at construction time without the need for multiple colonies. Objective weights are commonly used to combine information between objectives, either at construction time (in the case of multiple independent matrices) or during pheromone deposit (in the case of a single pheromone matrix). This paper focusses on the former. Similarly, heuristic factors may be treated separately or as an aggregate function. The selection of solutions to reward describes how the concept of elitism (a single-best ant depositing pheromone) is extended to MOACO. This distinguishes between allowing all of the non-dominated solutions to deposit pheromone and allowing only the best-per-objective to deposit. Alternatively, one could allow all ants to deposit pheromone relative to the quality of their solutions as in the traditional AS algorithm (i.e., no elitism).

Following the identification of three MOACO component categories (see above), Alaya et al. [1] propose a generic framework for defining MOACO implementations (deemed m-ACO). The m-ACO framework is parametrized by the number of colonies, and the number of pheromone trails. The number of colonies is either one more than the number of objectives or just 1, and the number of pheromone matrices is either equal to the number of objectives or set to 1. Ants will concentrate on a single pheromone matrix when constructing a complete solution, either the matrix associated with their home colony, or a randomly selected one. Where required, the multiple pheromone matrices may be combined into a single matrix via a linear sum (no objective weighting is used, or equivalently all weights are set to 1). An analogous technique is used for heuristic factors as well.

The preceding taxonomies are attempts to unify the existing MOACO literature into a single framework, not unlike the ACO metaheuristic described in Section 2.3. There are obvious differences in the components chosen to be highlighted by each taxonomy. However, it is believed that the reader's observation of all taxonomies combined will provide a firm understanding of the broad diversity and limited (but existent) constraints placed on the design choices of MOACO algorithms. The remainder of this section looks at a subset of individual MOACO implementations that showcase unique or original implementations of objective weight setting strategies.

Iredi et al.[17] implement and compare several variations of a MOACO algorithm for bi-criterion optimization problems. Variations include single or multiple colonies, two kinds of information sharing amongst multiple colonies (update by origin and update by region), and three rules for calculating objective weights. Aggregation of information is found both in the pheromone deposit rules and in the solution construction. The pheromone deposit takes two forms; in one ants only deposit pheromone in the colonies to which they belong. In the second form each colony is responsible for a section of the pareto front. Ants then update the pheromone in the colony which is responsible for the area of the pareto front in which their solution resides. This inter-colony pheromone deposit is considered a form of information aggregation since the individual colonies are shown to specialize in areas of the solution space. More explicit aggregation via the use of objective weights is found in the bi-criterion edge selection formula:

$$p_{ij,k} = \frac{[\tau_{ij,1}]^{\alpha\lambda_k} \cdot [\tau_{ij,2}]^{\alpha(1-\lambda_k)} \cdot [\eta_{ij,1}]^{\beta\lambda_k} \cdot [\eta_{ij,2}]^{\beta(1-\lambda_k)}}{\sum_{h \in N_{i,k}} [\tau_{ih,1}]^{\alpha\lambda_k} \cdot [\tau_{ih,2}]^{\alpha(1-\lambda_k)} \cdot [\eta_{ih,1}]^{\beta\lambda_k} \cdot [\eta_{ih,2}]^{\beta(1-\lambda_k)}} \tag{10}$$

where $[\tau_{ij,g}]^{\alpha\lambda_k}$ is the pheromone on edge$_{ij}$ associated with objective g weighted by the parameter α and the objective weight λ_k (analogous for η and β). The objective weight λ_k is different for each ant k, and is distributed uniformly across all ants using the following equation:

$$\lambda_k = \frac{k-1}{K/(C-1)} \tag{11}$$

where K is the number of ants and C is the number of colonies. Two variations on this rule are tested in [17], disjoint λ-intervals where each colony has a section of the range of lambda values, and overlapping λ-intervals similar to the disjoint rule, but with some overlap between 'adjacent' colonies. The λ-rule or linearly-distributed weighting concept is popular among weight setting strategies and is found in many other bi-objective ACO implementations. This is due to the notion that it evenly distributes the search priority of each objective amongst the population of ants. The down-side of this strategy is that it does not scale beyond bi-objective use in a straight-forward manner.

Cardoso et al.[7] propose a variant of the ACO heuristic for multi-objective network optimization known as MONACO. Information about each objective is kept in separate pheromone matrices and aggregated using *parameter-based* weightings at decision time using the equation:

$$p_{ij} = \begin{cases} \frac{[d_{jt}]^{-\alpha_0} \prod_{g=1}^{G} [\tau_{ij,g,t}]^{\alpha_g}}{\sum_{h \in N_i} [d_{ht}]^{-\alpha_0} \prod_{g=1}^{G} [\tau_{ih,g,t}]^{\alpha_g}} & \text{if } j \in N_i \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

where t is the destination node (specific to the network optimization domain), and d_{jt} is the Euclidean distance from node j to the destination node. Here α_g is the weight for objective g, provided as a parameter and remains constant for the entire run. This technique presumes some advanced knowledge about the priorities or qualities of each objective prior to the execution of the ACO algorithm in order to set appropriate values to the weights. This is considered 'fixed' information aggregation as described in [5]; however, the developed system is designed in the form of 'dynamic' information aggregation such that the weight values could be readily changeable [4].

Doerner et al. present an application of MOACO called Pareto Ant Colony Optimization (P-ACO) to the task of portfolio selection [8, 9]. The approach to aggregation of multiple pheromone matrices is via randomly selected objective weights for each ant. The objective weights are selected from a uniform distribution in the range [0,1). These are applied during the solution construction phase using the following variant of the edge selection formula to determine the probability of adding project i to the current portfolio x:

$$p_{i,k}(x) = \begin{cases} \frac{[\sum_{g=1}^{G} (\omega_{k,g} \cdot \tau_{i,g})]^{\alpha} \cdot [\eta_i(x)]^{\beta}}{\sum_{h \in N_k(x)} ([\sum_{g=1}^{G} (\omega_{k,g} \cdot \tau_{h,g})]^{\alpha} \cdot [\eta_h(x)]^{\beta})} & \text{if } i \in N_k(x) \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where $\omega_{k,g}$ is the weighting associated with objective g for ant k.

Barán and Schaerer [6] consider a 3-objective variant of the VRPTW by adapting the ACS algorithm. A single ant-colony is used on a single pheromone trail. Information from the best (non-dominated) solutions concerning two of the three objectives is aggregated as an unweighted product into the pheromone trail by each depositing ant, using the equation:

$$\tau_{ij} = \frac{\rho}{f_1(\psi) * f_2(\psi)}, \forall edge_{ij} \in \psi \tag{14}$$

where $f_g(\psi)$ is the fitness of solution ψ with regards to objective g. This single pheromone matrix is then used in conjunction with two desirability values, one for each of the same two objectives used in the pheromone. Thus the edge selection formula used by ant k is of the form:

$$p_{ij,k} = \begin{cases} \frac{\tau_{ij} \cdot [\eta_{ij,1}]^{\lambda_k \beta} \cdot [\eta_{ij,2}]^{(1-\lambda_k)\beta}}{\sum_{h \in N_{i,k}} \tau_{ih} \cdot [\eta_{ih,1}]^{\lambda_k \beta} \cdot [\eta_{ih,2}]^{(1-\lambda_k)\beta}} & \text{if } j \in N_{i,k} \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

where $\lambda_k = k/K$ with k being the current ant's index and K the total number of ants. Note that this is a simplified version (using a single ant colony) of the λ-rule described in the paper by Iredi et al. (see above). This gives the population of ants a linearly distributed view of the search space with regards to the desirability measures and a consistent view with regards to pheromone. Information about the third objective is not considered by the ants, and thus is not aggregated. Instead it is optimized as a by-product of the construction procedure.

Guntsch and Middendorf [15] extend Population-based ACO (PACO) [14] to multi-criteria optimization (PACO-MO). This extension performs a dynamic weight-based construction where the weights are repeatedly updated based on the qualities of solutions in the population. First, a probability distribution is calculated over each objective using the single-objective ACO decision rule (6). These probabilities are then combined via a weighted sum:

$$p_{ij} = \sum_{g}^{G} \omega_g \cdot p_{ij,g} \tag{16}$$

where $p_{ij,g}$ is the probability of selecting edge$_{ij}$ with respect to objective g using the single-objective ACO edge selection formula, and w_g is the weighting of objective g. Weights are calculated based on the relative quality of the solutions considered, formally:

$$\omega_g = \frac{1}{|P|} \sum_{k=1}^{|P|} \omega_g(\psi_k) \quad (17)$$

where $|P|$ is the size of the population (see [14] for a description of PACO population management), and $\omega_g(\psi_k)$ is the weight of objective g with respect to solution ψ_k, which is calculated as follows:

$$\omega_g(\psi_k) = \frac{r_g(\psi_k)}{\sum_g^G r_g(\psi_k)} \quad (18)$$

here, $r_g(\psi_k)$ is the rank of the kth ant's solution, ψ_k, with respected to objective g. Thus the individual solution weightings are calculated as the relative quality of one objective as compared to all other objectives. The end result of this is a highly dynamic form of global objective weighting. Objective weights are 'dynamic' in that a given weight will be recalculated many times throughout an experiment, and are 'global' in the sense that all ants in the population use the same weights at any given time.

Angus [3] later extends the PACO-MO algorithm. Among the alterations is the use of a single pheromone matrix to represent a ranking-based measure of the quality of a solution component (as opposed to several fitness-based measures). Individual heuristic matrices are maintained for each objective when available. Instead of calculating and maintaining objective preference weights, each ant is assigned a random-valued weight per objective. This is designed to allow varying degrees of heuristic exploitation per objective while sharing a single pheromone matrix. The edge selection formula used is adjusted accordingly:

$$p_{ij,k} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha} \cdot \prod_{g=1}^{G} [\eta_{ij}]^{\omega_{g,k} \cdot \beta}}{\sum_{h \in N_{i,k}} [\tau_{ih}]^{\alpha} \cdot \prod_{g=1}^{G} [\eta_{ih}]^{\omega_{g,k} \cdot \beta}} & \text{if } j \in N_{i,k} \\ 0 & \text{otherwise} \end{cases} \quad (19)$$

where $\omega_{g,k}$ is the weighting applied to objective g by ant k. This is similar to the technique used in [6], with random valued weights instead of linearly distributed ones.

To summarize, there are a large number of MOACO applications that use the concept of objective weights in a variety of different forms. There exists in the literature variations not only on the mechanism used to set these weights, but also in the subsequent use of the weights. The general form of the MOACO decision formula presented in Equation 9 can be modified to be made more general in light of the above implementations. Firstly, it can be modified to incorporate the notion of heterogeneous colonies of ants, where each ant may have a unique set of weights. A colony can be defined as a subset of the population of ants, typically associated with a single objective or region of the search space. The lack of a specific colony implementation is equivalent to the use of a single colony. Thus, to allow for heterogeneous colonies, each ant may be treated independently with shared constraints where appropriate. An additional modification to Equation 9 concerns the implementation-specific option to use the sum or product over the various values per objective. The updated general MOACO decision formula is presented in Equation 20.

$$p_{ij,k,t} = \frac{F_{a=0}^{A}([\tau_{ij,a}]^{\alpha_{a,k,t}}) \cdot F_{b=0}^{B}([\eta_{ij,b}]^{\beta_{b,k,t}})}{\sum_{h \in N_{i,k}} (F_{a=0}^{A}([\tau_{ih,a}]^{\alpha_{a,k,t}}) \cdot F_{b=0}^{B}([\eta_{ih,b}]^{\beta_{b,k,t}}))} \quad (20)$$

where F is a function that gives either the sum or product of its parameters depending on the current implementation.

An important distinction among the MOACO implementations presented above are the restrictions placed on the possible values of the objective weighting variables. Often cited as a means of facilitating diversification or directing the search towards desirable areas of the search space, MOACO algorithms impose a number of implicit and explicit constraints on the objective weights. To this end, the values of α and β in Equation 20 may be subject to a number of implementation dependent constraints. Examples of such constraints which may be inferred from the literature are presented in Table 1.[4] Note that there exist two types of constraints here: *initialization constraints*, and *adjustment constraints*. Initialization constraints (denoted $\alpha_{a,k,0}$) are applied once at the beginning of execution, whereas adjustment constraints (denoted $\alpha_{a,k,t}$) are applied as a daemon action step in the MOACO metaheuristic once per iteration. Weight values in the absence of any adjustment constraints are defined as constant values upon initialization. Therefore adjustment constraints may be considered optional while initialization constraints may not.

Table 1: Example weighting constraints

Constraint	Description
$\alpha_{a,k,0}$ = x	Explicit setting of weight, where x is a parameter.
$\alpha_{a,k,0} = \frac{k-1}{K/(C-1)}$	'λ-rule' from [17].
$\alpha_{a,k,0}$ = U[0,1)	Uniform random number [8].
$\alpha_{a,k,0} = 1\ \forall a \in A$	Unweighted [1].
$\alpha_{a*,k,0} = 1$, $\alpha_{a,k,0} = 0$ $\forall a \neq a* \in A$	Single-objective focus [1].
$\alpha_{a,k,t} = \frac{r_a(\psi_k)}{\sum_a^A r_a(\psi_k)}$	Rank-based dynamic weight setting from [6].
$\alpha_{a,k,t+1} = \alpha_{a,k,t} + U[-0.1, 0.1)$	Weights vary by a small random shift at each time step.
$\alpha_{a,k,t} = \alpha_{a,k*,t}\ \forall k \in K$	Homogeneous weightings for all ants in a colony or population.

It should be noted that while many of the reviewed implementations use both the traditional α and β symbols to weight the relative importance of pheromone and heuristic information respectively as well as an *additional* symbol to control the objective weights (e.g., λ or ω), the generalized decision formula uses only α and β. This is not a limitation as constraints can be readily designed to contain both forms of information. For example, $\alpha_{a,k} = x * U[0,1)$ could be considered equivalent to $\alpha * \omega$ in a case where the objective

[4]Shown are constraints on the α value. An analogous case may be made for β values as well.

weight ω is set randomly and the pheromone weight α is equal to x.

This constraint-based form of representation represents a novel interpretation of the MOACO literature reviewed and serves to elucidate the experimental study of objective weight setting strategies described in the remainder of this paper.

4. EXPERIMENTAL SETUP

4.1 Implementation Details

The developed algorithm, Algorithm 1, was inspired from the large number of MOACO algorithms found in the literature with varying implementations of the design choices, in particular, relating to weight setting strategies. To address this the MOACO algorithm developed here is designed to be adjustable enough to suit various implementation scenarios.[5]

The MOACO implementation presented here is based upon the AS implementation of the ACO metaheuristic. Conceivably, more advanced implementations (such as the MaxMin Ant System (MMAS)[24] or the Ant Colony System (ACS)[10]) could be extended using the MOACO principles, however the simplicity of the base algorithm here highlights the contribution of the MOACO implementation decisions. It is for the same reason that no local search is used in the implementation despite the fact that general ACO knowledge suggests the contrary.

The implemented system incorporates a multi-objective approach by using an independent pheromone matrix for each objective under consideration. Ants coordinate their use of multiple pheromone matrices by means of a list of objective weights, where each weight specifies a given ant's sensitivity to a single pheromone matrix (and therefore to a single objective). This effectively decouples the ants' solution construction behaviour from their prescribed strategy for coping with multiple objectives. The latter is determined by the setting of weight-constraints, while the former is a fixed set of instructions utilizing whatever values the objective weights are currently set to.

The pseudo-code of the implemented MOACO algorithm is shown in Algorithm 1. It should be noted that for the purposes of generality, the pseudo-code associates pheromone and desirabilities with the edges in the graph, however it is a simple adjustment to associate these with the vertices instead (as is the case in solving the MOKP).

Initialization consists of creating and instantiating a pheromone matrix and heuristic matrix for each objective. A population of ants is created and initialized analogous to the procedure in a single objective set-up, with the exception that each ant is given an array of weights. The number of weights is dependent on the number of objectives in the current problem as well as the user-specified use of weights for each pheromone matrix, each heuristic matrix, both, or neither (i.e., G weights, 2xG weights, or 0 weights). The values for the weight lists are determined by the initialization constraints.

[5]The implementation is available online: http://people.scs.carleton.ca/∼arunka/research/moaco/

Algorithm 1 Pseudo-code of the MOACO algorithm

```
{Initialization}
for all Objectives g do
  for all Pairs of vertices i, j do
    desirability[g][i][j] ← aPrioriKnowledge()
    pheromone[g][i][j] ← InitialPheromoneValues()
  end for
end for
for all Ants k in Population do
  for all Objectives g do
    k.weight[g] ← InitializationContraints()
  end for
end for
{Main Loop}
while !Termination do
  {Construct Solutions}
  for all Ants k in Population do
    i ← 0
    while k.solution is incomplete do
      for all Vertices j in Neighbourhood(i) do
        Compute p(i, j) {Equation 20}
      end for
      k.solution[i] ← RouletteSelection()
      i ← i + 1
    end while
    EvaluateSolution(k.solution)
  end for
  {Select Best Solutions}
  Candidates ← Population ∪ NonDominatedSolutions
  NonDominatedSolutions ←Ø
  for all Ants k in Candidates do
    if k.solution is NonDominated then
      NonDominatedSolutions ← k
    end if
  end for
  {Adjust Weights, if applicable}
  for all Ants k in Population do
    for all Objectives g do
      k.weight[g] ← AdjustmentConstraints()
    end for
  end for
  PheromoneEvaporation() {Equation 21}
  for all Ants k in NonDominatedSolutions do
    Update Pheromone {Equation 22}
  end for
end while
return NonDominatedSolutions
```

Following initialization, the main loop proceeds by first constructing solutions. A given ant will add vertices to its path based on the MOACO decision formula (Equation 20). Next, the solutions are evaluated. This is done for each objective resulting in an array of fitness values. Based on these fitness values the solutions are then compared to one another in order to determine the Pareto front. Those solutions that are non-dominated are cloned and and kept as the current best individuals. Each ant in the population then undergoes a weight-update based on the adjustment constraints (if any).

Once all of the weights have been updated, each ant in the colony deposits pheromone on the vertices it visited. The pheromone evaporation and deposit is calculated independently for each objective yielding Equations 21 and 22.

$$\tau_{ij,g} = (1-\rho)\tau_{ij,g} \quad (21)$$

$$\tau_{ij,g} = \tau_{ij,g} + (1 - \frac{1}{f_g(\psi_k)}) \quad (22)$$

where $\tau_{ij,g}$ is the amount of pheromone on $edge_{i,j}$ for objective g, $f_g(\psi_k)$ is the fitness of the k^{th} ant's solution with regards to objective g, and ρ is a parameter that controls the rate of evaporation. Note that Equation 22 is applied once for every ant in the pareto front.

This main search loop repeats for a set number of iterations (or until some other termination condition is satisfied). Each time through the loop, the ants construct their solutions from scratch, however the weight values for each ant in the population persist from iteration to iteration. In this way the ants have an opportunity to utilize a limited form of memory with which to learn over the duration of an entire experiment.

4.2 Parameters

To examine the performance of the implemented MOACO configurations, the algorithm was run on 12 instances of the MOKP ranging in size from 100-750 objects, with 2, 3, and 4 objectives. Problem instances are described in a consistent format in the following sections. As an example, the instance k.500.3 represents a MOKP problem instance with 500 items per objective and 3 objectives.

In order to highlight the contribution of the information aggregation scheme no local search is used, and the underlying ACO algorithm is the Ant System unless otherwise specified.

Experiments are run for 1000 generations in order to give ample time for convergence. In all runs the population size is set to 100 ants, and the evaporation rate is set to 0.1. The parameters of the algorithm were tuned experimentally and thus may be dependent on the problem at hand. Constraint-specific parameters are described below their respective descriptions in the next section.

4.3 MOACO elements of study

The remainder of this paper is focussed on examining the impacts of weight-setting design decisions, in terms of their effect on solution quality and search behaviour. The experiments performed are intended to explore the differences in constraints gleaned from the MOACO literature as well as several novel constraints.

A number of experiments will be run using initialization constraints and adjustment constraints to examine features of working with objective weights. The following constraints will be examined:

Unweighted - No weights are used. Equivalently,

$\alpha_{a,k,0} = 1 \ \forall a, k$,

where a is the pheromone matrix index, and k is the ant index.

Linear - Weights are evenly distributed along the line x = 1-y. (Bi-objective only).

$\alpha_{0,k,0} = \frac{k-1}{K}$,

$\alpha_{1,k,0} = 1\text{-}\alpha_{0,k,0}$,

where K is the number of ants, set to 100.

Random - Weights are drawn randomly from a uniform distribution.

$\alpha_{a,k,0} = \text{U[min,max)}$,

where min and max are the boundaries of the weights, set to 0 and 2 respectively.

Random Shift - Weights are shifted in a random direction/amount. Values are bounded.

$\alpha_{a,k,t} = \alpha_{a,k,t-1} + U[-s, s)$,

$min \leq \alpha_{a,k,t} \leq max$,

where s is a parameter that controls the size of random shift, set to 0.1, and t is a given time-step.

Relative Quality - Weights are adjusted based on their relative rank within the population.

$\alpha_{a,k,t} = \frac{r_a(\psi_{k,t})}{\sum_a^A r_a(\psi_{k,t})}$,

where $r_a(\psi_{k,t})$ is the rank of the solution constructed by ant k at time-step t on objective a compared to all ants.

Equal Plus/Minus - Weights vary, but remain close along each dimension.

$\alpha_{0,k,t} = \alpha_{0,k,t-1} + U[-s, s)$, and

$\alpha_{a,k,t} = \alpha_{0,k,t} + U[-r, r) \ \forall a \in A$,

$min \leq \alpha_{a,k,t} \leq max$,

where r is a parameter that controls the spread of solutions around the x = y = z =... line, below it is considered at values of 0 and 0.1.

Random Cluster - Weights are shifted towards the closest weight-set in the pareto-front with some probability otherwise moved randomly.

$$\alpha_{a,k,t} = \begin{cases} \alpha_{a,k,t-1} + \delta * (\alpha_{a,p',t-1} - \alpha_{a,k,t-1}) + \phi, \text{ or} \\ \alpha_{a,k,t-1} + U[-s, s) \text{ with probabilty q} \end{cases},$$

$min \leq \alpha_{a,k,t} \leq max$,

where δ is the step-size taken towards a cluster's centre, ϕ is a small amount of noise, and p' is the nearest weight that is in the current pareto front. Here δ is set to 0.75, ϕ is set to 0.05, s is set to 0.3, and q is set to 0.8.

Global - All ants share a single global set of objective weights.

$\alpha_{a,k,t} = \alpha_{a,0,t} \ \forall k \in K$,

where $\alpha_{a,0,t}$ is set according to some other constraint(s), below it is set using the Random Shift constraint.

In addition to these, the experimental section begins by examining an important underlying concept that may be easily overlooked in the foregoing descriptions: the actual effect of a weight on the search behaviour of an artificial ant.

5. RESULTS AND DISCUSSION

This section presents the experimental results of this paper accompanied by succinct discussion on the same, but first, a preliminary note. The experiments performed here are not intended to establish a benchmark of performance. The intent is instead to expand upon the above review of weight setting strategies and proposal of constraint-based specifications. This is achieved through experiments that examine the impacts of various weight-related design choices. It is the hope of this author that such experiments will ultimately be more valuable for instigating well-established implementation guidelines in the MOACO community.

5.1 General Objective Weighting

Prior to the comparison of various weight setting strategies, the effect of objective weights themselves must be established. While this may seem trivial, it underlies the practicality of the remainder of this work (as well as much of the foregoing literature). The assumption made when using objective weights is that by biasing an ant's preference for one objective-based pheromone, the ant will stress the importance of that objective in its solution construction. So it will more likely create solutions that have high utility in one objective potentially at the cost of allowing lower utility in the non-biased objectives. Therefore various mechanisms of weight manipulation can lead to a variety of search techniques that use an ant's objective preference to explore specific niches of the search-space.

In order to test this assumption an experiment was generated that would bias the ants towards a specific objective. To this end, the following constraints were used:

- $\alpha_{a,k,0} = 0, \ \forall a \in A : a \neq a*$,
- $\alpha_{a*,k,0} = 1$, where $a*$ is the biased objective of the current experiment.
- $\alpha_{a*,k,t} = \alpha_{a*,k,t-1} + U[-0.1, 0.1]$,
- $0 \leq \alpha_{a*,k,t} \leq 2$

When run on a problem of size 500 with 3 objectives, and averaged over 30 runs per each selected objective-bias, the the results in Table 2 were observed. As can be readily seen (and easily verified on additional problem instances), the results obtained under objective bias are greater than those obtained without. The differences between biased and unbiased results is statistically significant given a 95% confidence interval, while the differences between non-biased results for a given objective are not. Therefore it holds that objective weights can be used as a means of biasing search towards areas of the search space. The degree and the way in which this can be done beneficially is the focus of the remainder of experiments.

5.2 Linear Weights

The practical intention of using a distribution of static objective weights is to spread the search along the entire pareto front, rather than focussing the search at a single point of trade-off. The purpose of this first experiment is to gauge to what extent this spread is achieved using a simple constraint taken from the literature. As seen in Figure 1, the linear distribution constraint creates a uniform gradient between weighting entirely on one objective and entirely on the other. Each ant explores a slightly different range of the trade-off between objectives. This translates to a search strategy as seen in Figure 2. Depicted here is a superposition of the 2-dimensional fitness of every individual in the population as well as the pareto front at every iteration over an entire run. This illustrates the sampling distribution performed during search under the specified weighting constraints. Equivalently, Figure 3 presents the sampling distribution for an example run of the unweighted experiment. Both of these figures are characteristic of all the experiments performed using their respective parameters. Through simple observation of these figures, the distributional advantage using the linear weights seems apparent. The samples taken using objective weights form a wider spread along the trade-off surface, while the unweighted samples converge around a point that is near the centre of the trade-off surface.

Table 2: Objective bias by mean pareto-front fitness per objective

Bias\Mean	Objective 1	Objective 2	Objective 3
Objective 1	**18193.80**	16543.18	16533.95
Objective 2	15989.49	**18202.50**	16495.01
Objective 3	15970.90	16471.72	**18724.91**

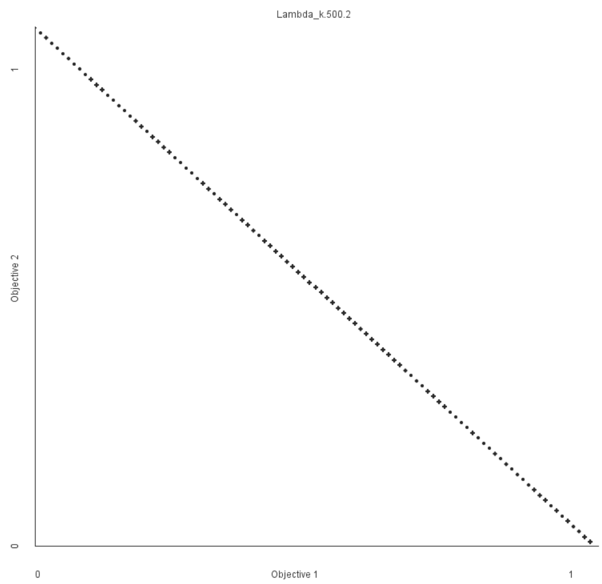

Figure 1: Objective weight distribution given the linear distribution initialization constraints.

These observations are corroborated experimentally for small instances as seen in Table 3. Here the coverage values are as explained in Section 2.1.1, the hypervolume column refers to the volume attained by one set that is not attained by the other, and the univariate column refers to the percentage of space for which one set of results is statistically better than the other given a 95% confidence interval.

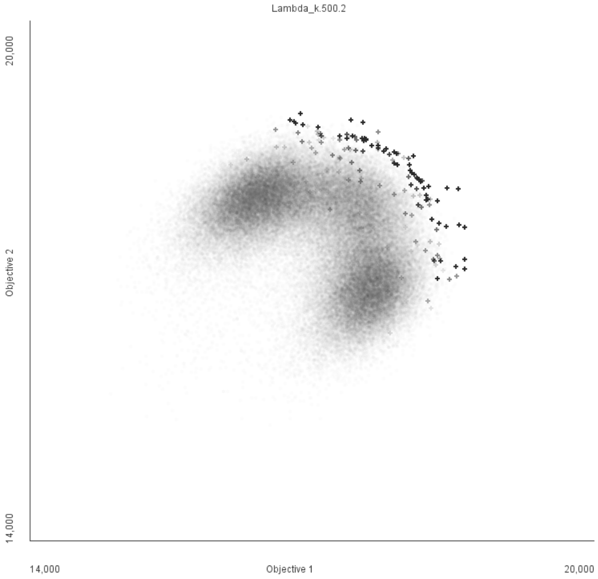

Figure 2: Fitness-space sampling distribution using linearly distributed objective weights over the course of 1000 iterations on the k.500.2 problem instance. Circular points represent the population samples, while plus-shaped points represent the pareto front.

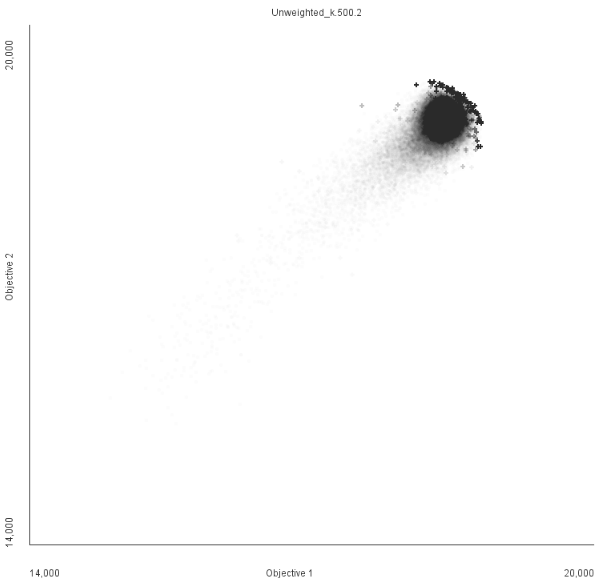

Figure 3: Fitness-space sampling distribution using no objective weights.

Table 3: Comparison of unweighted versus linearly distributed objective weights.

Instance	Constraint	Coverage	Hypervol.	Univar.
k.100.2	Unweighted	0.472	$4.46\text{x}10^3$	32.2
	Linear	0.214	$4.56\text{x}10^5$	58.0
k.250.2	Unweighted	0.921	$7.35\text{x}10^5$	74.0
	Linear	0.0	$5.08\text{x}10^5$	22.2
k.500.2	Unweighted	0.993	$1.12\text{x}10^7$	100.0
	Linear	0.0	$9.38\text{x}10^4$	0.0
k.750.2	Unweighted	0.997	$2.91\text{x}10^7$	100.0
	Linear	0.0	$9.36\text{x}10^4$	0.0

Considering first only the smallest bi-objective instance, a comparison of the experimental results of both constraints using the coverage measure indicates that, on average, the pareto fronts from unweighted experiments dominate more points in the pareto fronts of the linear weighted experiments, than vice versa. However, a comparison of the same results using the hypervolume metric indicates that on average the linear weighted pareto fronts dominate a larger volume of the fitness space for the smallest instance. This suggests that while the unweighted experiment finds a high density of non-dominated solutions at an equilibrium point on the pareto front, it does so at a cost of limited exploration away from that point. This implies that the static distribution of objective weights serves its design goal. That is, it effectively maintains a broad spectrum of objective biases in order to diversify the solution sampling, and thus the pareto front. However, as the problem size scales up, the advantage of the linear objective-weight distribution appears to fade. This issue is addressed in the following section.

5.3 Adjusted Linear Weights

The limited quality attained by the linearly distributed objective weights is likely due to the limited range of weight values. Using the definition of linearly distributed weights as presented in Section 4.3, all weights are in the range [0,1]. That means that the objective weights are actually reducing the importance of each pheromone owing to sublinear amplification. This also means that the desirability information (η) becomes more influential as a consequence. A slight modification to the constraints tests this hypothesis by adding 1 to each weight in order to ensure that all weights are in the range [1,2]. Formally:

- $\lambda_k = \frac{k-1}{K}$,
- $\alpha_{0,k,0} = 1 + \lambda_k$,
- $\alpha_{1,k,0} = 2 - \lambda_k$.

The results of this modification as compared to the unmodified and unweighted constraints are presented in Tables 4 and 5 respectively. The modified constraints appear to be more scalable, with the comparison between unmodified becoming more and more starkly contrasted as the problem size increases. Similarly, the results of the modified linear constraints remain competitive with the unweighted constraint, and surpass it in quality for the largest problem instance. Unfortunately, the limited applicability of the linear constraints to bi-objective instances limits the depth of

Table 4: Comparison of linearly distributed versus modified linearly distributed objective weights

Instance	Constraint	Coverage	Hypervol.	Univar.
k.100.2	Linear	0.369	4.06x10^5	59.8
	Linear+1	0.359	9.71x10^3	14.0
k.250.2	Linear	0.0	7.04x10^5	19.3
	Linear+1	0.906	8.30x10^5	71.2
k.500.2	Linear	0.0	7.55x10^4	0.0
	Linear+1	0.996	1.12x10^7	100.0
k.750.2	Linear	0.0	0.0	0.0
	Linear+1	1.0	3.74x10^7	100.0

Table 5: Comparison of unweighted versus modified linearly distributed objective weights

Instance	Constraint	Coverage	Hypervol.	Univar.
k.100.2	Unweighted	0.470	9.78x10^4	61.2
	Linear+1	0.170	1.54x10^5	29.3
k.250.2	Unweighted	0.496	5.41x10^5	27.5
	Linear+1	0.340	4.41x10^5	0.0
k.500.2	Unweighted	0.345	2.49x10^6	43.7
	Linear+1	0.369	2.56x10^6	48.1
k.750.2	Unweighted	0.096	2.30x10^6	0.0
	Linear+1	0.741	1.08x10^7	58.8

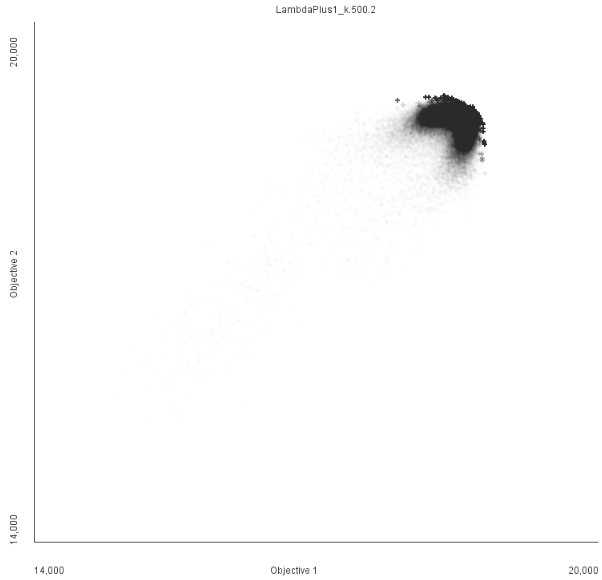

Figure 4: Fitness space sampling distribution using the modified linear objective weights.

comparison along these lines. This issue will be returned to with later results.

The fitness sampling distribution of the modified constraint is presented in Figure 4. The modified constraint appears to take on a behaviour that is more concentrated than the previous linearly distributed weights, and yet more distributed than the unweighted samples. This reflects the presented results in that the modified constraints prove more scalable than the original constraints, however achieve results that are generally comparable to the unweighted on most instances.

Interestingly, the modified linear constraint appears to be outperformed, on the smallest instance, by the unmodified constraint. This suggests a problem dependent relationship with the use of objective weight constraints, as the smallest instance appears to be more tractable with a less diverse search technique.

The greatest limitation of the linearly distributed constraints is that they are defined as bi-objective constraints only. The remainder of constraints considered are applicable to an arbitrary number of objectives.

5.4 Randomly Initialized Weights

As a baseline test of an objective weight setting strategy that is applicable to any number of dimensions, consider the Random constraints as presented in Section 4.3. With this, all weights are initialized to values selected randomly from a uniform distribution on the range [0,2][6] for each objective.

This simple-to-implement constraint samples the objective weights as seen in Figure 5, and produces a fitness-space sampling distribution as seen in Figure 6. Despite the stark contrast in the objective weight distribution, the fitness-space distribution is reminiscent of that of the modified linear objective weighting scheme (Figure 4). This suggests that any static distribution of objective weights can achieve a similar spread, provided that the weight values are allowed to differ from one another, and obtain values greater than one. Slight differences noticeable in the amount of sampling done farther from the pareto front (closer to the origin) illustrate the effect of allowing values between 0 and 1 in the randomly initialized weight values, but not the linearly distributed ones. Another comparable result is noticed in the scalability of the randomly initialized objective weights as compared to the results of the unweighted constraint. The results of the random constraint according to all three metrics improve, with respect to the results of the unweighted constraint, as the complexity of the problem increases. It should also be noted that the dominant hypervolume of the randomly initialized weight results is greater than that of the unweighted results in all but one instance, indicating that a broader sampling of the pareto front occurs given diversity in the weight values.

These results suggest that a broad variety of biased samplings, as provided by the use of some weight distribution (albeit a random one), is beneficial to the search for quality solutions in large-sized higher-dimensional multi-objective problems.

5.5 Randomly Shifting Weights

[6]While extensive parameter tuning is beyond the scope of this paper, the range [0,2] was determined experimentally. This range of weights enables both amplification and suppression behaviours. The combination of both behaviours produces superior results in sample experiments when compared to ranges that exhibit either behaviour exclusively (e.g., [0,1] and [1,2] respectively).

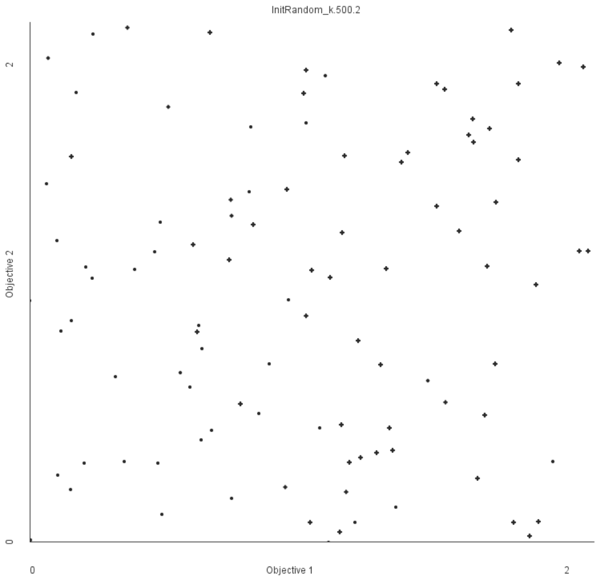

Figure 5: Objective weight distribution using randomly initialized objective weights.

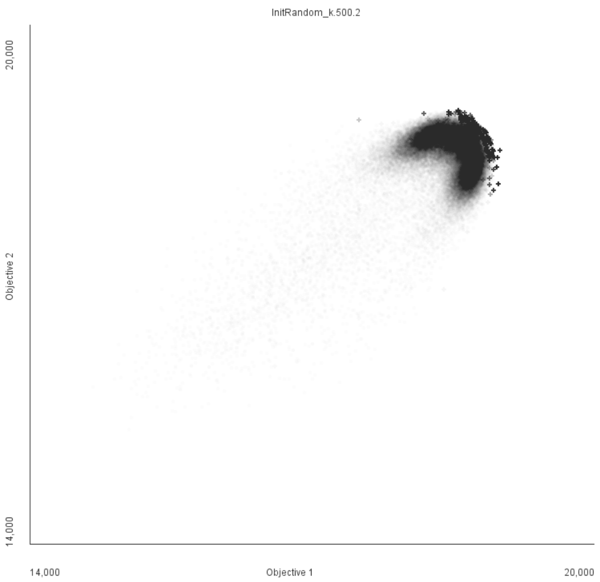

Figure 6: Fitness-space sampling distribution using randomly initialized objective weights.

Table 6: Comparison of unweighted versus randomly initialized objective weights

Instance	Constraint	Coverage	Hypervol.	Univar.
k.100.2	Unweighted	0.331	$4.17\text{x}10^4$	11.7
	Random	0.274	$2.35\text{x}10^5$	34.0
k.250.2	Unweighted	0.641	$3.53\text{x}10^5$	39.7
	Random	0.188	$6.22\text{x}10^5$	17.6
k.500.2	Unweighted	0.589	$2.50\text{x}10^6$	62.0
	Random	0.165	$1.99\text{x}10^6$	28.0
k.750.2	Unweighted	0.352	$3.51\text{x}10^6$	41.8
	Random	0.419	$8.05\text{x}10^6$	48.6
k.100.3	Unweighted	0.168	$3.81\text{x}10^8$	6.6
	Random	0.521	$1.69\text{x}10^9$	61.4
k.250.3	Unweighted	0.048	$1.95\text{x}10^9$	0.0
	Random	0.790	$2.21\text{x}10^{10}$	65.7
k.500.3	Unweighted	0.037	$3.02\text{x}10^9$	0.0
	Random	0.839	$2.31\text{x}10^{11}$	99.3
k.750.3	Unweighted	0.008	$1.18\text{x}10^9$	0.0
	Random	0.894	$1.08\text{x}10^{12}$	94.7
k.100.4	Unweighted	0.146	$9.13\text{x}10^{11}$	0.0
	Random	0.487	$6.60\text{x}10^{12}$	100.0
k.250.4	Unweighted	0.018	$6.63\text{x}10^{12}$	0.0
	Random	0.849	$3.46\text{x}10^{14}$	100.0
k.500.4	Unweighted	0.002	$1.44\text{x}10^{11}$	0.0
	Random	0.958	$7.81\text{x}10^{15}$	100.0
k.750.4	Unweighted	0.000	$2.14\text{x}10^{12}$	0.0
	Random	0.975	$4.86\text{x}10^{16}$	100.0

The purpose of this next experiment is to test whether or not the property of weights being dynamic has an effect on the search when compared to static weights. To this end, the Random Shift constraints are employed, and the results compared to the randomly initialized static weight results.

The objective weight sampling distribution corresponding to randomly shifting weights is presented in Figure 7. Although this distribution is noticeably different as compared to the static distribution in Figure 5, the fitness-space sampling distribution, presented in Figure 8, appears nearly identical to its static counterpart, seen in Figure 6. To the same end, there does not appear to be any clear distinction between constraints in terms of the comparison results presented in Table 7. In over half of the instances considered the univariate metric indicates there is no statistically significant advantage to either set of constraints. In the remaining instances, the significant difference is on very small portions of the pareto front and is not consistently in favour of either constraint.

These results suggest that there is no intrinsic advantage, in terms of quality, derived from the property of having dynamic weights. Intuitively, the instantaneous distribution of objective weights using either a dynamically or statically random constraint will be a uniform random distribution over the values [0,2] for each objective. Thus, the solutions constructed by each ant will use a random weighting on the pheromone, and the solutions constructed by the population of ants will reflect the use of a uniform random distribution of weights. Therefore it should follow that so long as the instantaneous distributions of both constraints are

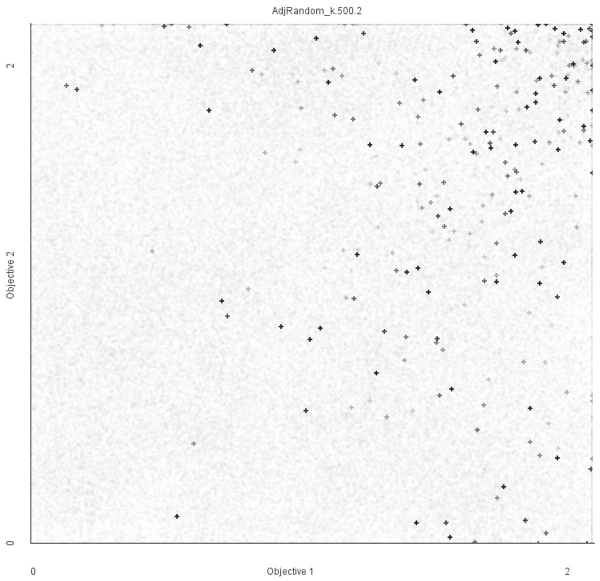

Figure 7: Objective weight sampling distribution using the random shift constraints.

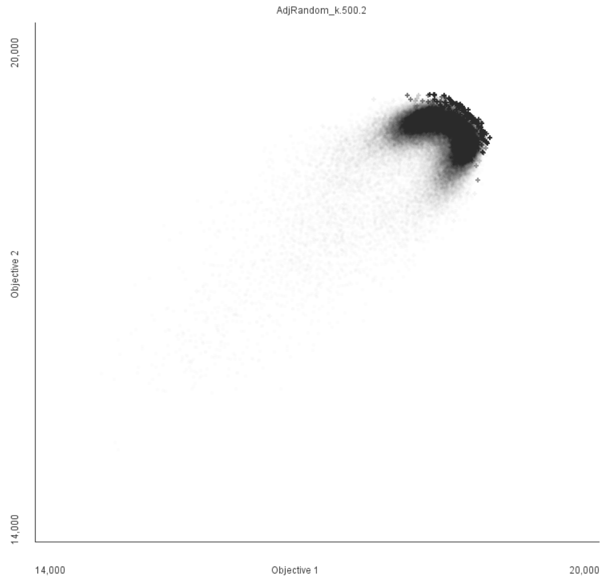

Figure 8: Fitness-space sampling distribution using randomly shifting objective weights.

comparable, the search behaviours should remain relatively consistent.

5.6 Relative Quality-based Weights

The Relative Quality constraint, as presented in Section 4.3, is one means of extending the linearly distributed objective weights beyond the bi-objective instances. The interest here concerns verifying the scalability results of the linear weights, as well as verifying the lack of impact of dynamically shifting weight values.

This constraint achieves a linear distribution in two dimensions (triangular plane in three dimensions, and so on) by ranking a given ant's performance on each objective against all other ants, and then setting the weight for a given objective proportionally to the total sum of all ranks achieved by that ant.[7] It is important to note that this requires the additional overhead of ranking all of the ants on each objective. Similar to the linearly distributed weights examined earlier, the results presented here are of a modified constraint such that all weights are within the range [1,2] instead of [0,1].

A difference between the linearly distributed initialization constraint and the relative quality constraint is that the latter enables each weight to change over time. However, the results of the previous experiment suggest that the property of dynamic weights should not impact the solution quality, so long as the instantaneous weight distributions are similar. This finding is supported by the bi-objective results in Table 8. The relative quality dynamic weights are comparable to the static linear weights. Considering the univariate column, there is little statistically significant difference in all

[7]Other forms of linear constraint are conceivable, the Relative Quality constraint used here was adapted from the existing literature, see [15]

Table 7: Comparison of randomly initialized objective weights versus randomly shifting objective weights

Instance	Constraint	Coverage	Hypervol.	Univar.
k.100.2	Random	0.338	1.02×10^{5}	6.0
	Rand. Shift	0.339	1.01×10^{5}	12.0
k.250.2	Random	0.425	6.14×10^{5}	0.0
	Rand. Shift	0.439	4.09×10^{5}	0.0
k.500.2	Random	0.386	1.92×10^{6}	0.0
	Rand. Shift	0.432	1.95×10^{6}	0.0
k.750.2	Random	0.421	4.56×10^{6}	4.3
	Rand. Shift	0.437	2.96×10^{6}	0.0
k.100.3	Random	0.418	5.57×10^{8}	13.8
	Rand. Shift	0.286	5.10×10^{8}	0.0
k.250.3	Random	0.280	4.83×10^{9}	0.0
	Rand. Shift	0.428	7.62×10^{9}	14.1
k.500.3	Random	0.252	4.61×10^{10}	0.0
	Rand. Shift	0.511	4.76×10^{10}	13.7
k.750.3	Random	0.358	1.15×10^{11}	0.0
	Rand. Shift	0.395	1.22×10^{11}	0.0
k.100.4	Random	0.303	2.14×10^{12}	0.0
	Rand. Shift	0.320	2.17×10^{12}	0.0
k.250.4	Random	0.338	6.68×10^{13}	0.0
	Rand. Shift	0.334	6.51×10^{13}	0.0
k.500.4	Random	0.256	7.82×10^{14}	0.0
	Rand. Shift	0.460	7.64×10^{14}	0.0
k.750.4	Random	0.290	3.83×10^{15}	0.0
	Rand. Shift	0.409	3.37×10^{15}	0.0

Table 8: Comparison of linearly distributed versus relative quality-based objective weights

Instance	Constraint	Coverage	Hypervol.	Univar.
k.100.2	Linear	0.353	1.37×10^5	0.6
	Relative	0.349	1.15×10^5	0.2
k.250.2	Linear	0.376	7.24×10^5	0.0
	Relative	0.410	8.59×10^5	0.0
k.500.2	Linear	0.288	1.66×10^6	0.0
	Relative	0.496	2.56×10^6	49.8
k.750.2	Linear	0.513	4.48×10^6	4.8
	Relative	0.324	3.17×10^6	0.0

but half of the pareto front of one instance. The coverage and hypervolume results are likewise similar.

Returning to the concept of scalability, Table 9 demonstrates, as before, that the advantage of objective weights over unweighted experiments increases with the problem complexity. One possible explanation for this, is that as the search space increases in size, the pareto front expands as well. So, the advantage of spreading the search along the pareto front becomes more pronounced as the problem complexity increases.

Table 9: Comparison of unweighted versus relative quality-based objective weights

Instance	Constraint	Cover.	Hypervol.	Univar.
k.100.2	Unweighted	0.449	9.26×10^4	34.5
	Relative	0.190	1.27×10^5	20.0
k.250.2	Unweighted	0.466	5.42×10^5	6.0
	Relative	0.361	5.76×10^5	0.0
k.500.2	Unweighted	0.260	1.56×10^6	33.2
	Relative	0.492	2.52×10^6	49.7
k.750.2	Unweighted	0.118	2.61×10^6	18.7
	Relative	0.657	9.76×10^6	68.1
k.100.3	Unweighted	0.120	5.06×10^8	16.9
	Relative	0.574	1.12×10^9	56.8
k.250.3	Unweighted	0.015	4.87×10^8	0.0
	Relative	0.927	2.03×10^{10}	99.4
k.500.3	Unweighted	0.003	1.08×10^9	0.0
	Relative	0.971	2.46×10^{11}	100.0
k.750.3	Unweighted	0.002	1.42×10^8	0.0
	Relative	0.982	1.01×10^{12}	100.0
k.100.4	Unweighted	0.070	1.05×10^{12}	0.0
	Relative	0.668	4.07×10^{12}	100.0
k.250.4	Unweighted	0.011	1.62×10^{13}	0.0
	Relative	0.891	2.66×10^{14}	100.0
k.500.4	Unweighted	0.003	7.13×10^{12}	0.0
	Relative	0.979	6.71×10^{15}	100.0
k.750.4	Unweighted	0.000	2.21×10^{12}	0.0
	Relative	0.997	4.02×10^{16}	100.0

5.7 Constrained-Difference Weights

As demonstrated in the results of the previous sections, an advantage to using objective weights over an unweighted approach is the ability to direct the search along more of the pareto front by diversifying the weight values. It is useful to know, then, how diverse the objective weights must be from one another in order to effect a change in the search behaviour. In order to test the sensitivity of the search results to changes in objective weight values, the 'Equal Plus/Minus' constraint set is used.

Table 10: T-test results of Equal versus Equal +/- 5%, significant results shown in bold.

Instance	Obj. 1	Obj. 2	Obj. 3	Obj. 4
k.100.2	0.398	0.985	–	–
k.250.2	0.318	0.312	–	–
k.500.2	**0.008**	**0.019**	–	–
k.750.2	0.114	0.562	–	–
k.100.3	0.704	0.155	0.117	–
k.250.3	0.067	$\mathbf{9.5 \times 10^{-4}}$	$\mathbf{1.4 \times 10^{-4}}$	–
k.500.3	$\mathbf{2.6 \times 10^{-4}}$	$\mathbf{4.4 \times 10^{-6}}$	$\mathbf{7.7 \times 10^{-4}}$	–
k.750.3	**0.010**	$\mathbf{6.7 \times 10^{-5}}$	$\mathbf{1.1 \times 10^{-4}}$	–
k.100.4	0.542	$\mathbf{8.8 \times 10^{-4}}$	0.225	0.067
k.250.4	0.079	$\mathbf{1.9 \times 10^{-5}}$	**0.009**	$\mathbf{7.9 \times 10^{-4}}$
k.500.4	**0.028**	$\mathbf{3.8 \times 10^{-4}}$	$\mathbf{5.6 \times 10^{-6}}$	$\mathbf{1.6 \times 10^{-6}}$
k.750.4	**0.002**	$\mathbf{2.1 \times 10^{-8}}$	$\mathbf{2.9 \times 10^{-9}}$	$\mathbf{5.1 \times 10^{-7}}$

A comparison was conducted between constraints that allow zero variance in objective weights (i.e., Equal Plus/Minus with r = 0), and the same with a small variance allowed (i.e., r = 5% of the range of weight values). Each variant was run 30 times for each instance and the standard deviations of the pareto front values for each objective was measured. A two-tailed t-test with a 95% confidence interval was calculated between each corresponding set of standard deviations, with the null hypothesis being that the small increase in variance allowed in the constraint has no effect on the standard deviation (and therefore the breadth of the pareto front) along each objective.

The results of the t-tests are presented in Table 10. The percentage of instance-objective pairs for which the change in standard deviation is statistically significant (≤ 0.05) is approximately 61% just by allowing a variance of +/- 5% of the range. This increases to 92% significant changes if the variance in increased slightly more to 10% of the range.

This shows that the breadth of results obtained is highly sensitive to small changes in the limits placed on the values of the objective weights. The concentration of rejected null-hypotheses in Table 10 towards the more complex instances suggests that this sensitivity is more influential when there is a larger search space.

5.8 Shared Global Weights

The choice between using global or individual objective weights is a prominent design decision in the MOACO literature. Individual weights are of the form presented throughout the foregoing results of this paper, where each ant constructs its solution using an independent set of objective weights. Alternatively, global weights are a single set of objective biases that all ants share. While a single global weight is conceptually easier and more efficient to implement (requiring only a single computation of shared decision information for an entire population of ants), the use of individual weights enables a diversity of simultaneous biases. A global weight constraint is employed in order to gauge the utility of either side of this design decision.

Table 11: Comparison of a single global weight versus separate weights for each individual ant

Instance	Constraint	Coverage	Hypervol.	Univar.
k.100.2	Global	0.348	$2.58x10^{5}$	24.9
	Individual	0.267	$1.73x10^{5}$	6.2
k.250.2	Global	0.198	$1.40x10^{6}$	29.1
	Individual	0.452	$1.02x10^{6}$	37.5
k.500.2	Global	0.153	$5.20x10^{6}$	0.0
	Individual	0.430	$5.95x10^{6}$	38.7
k.750.2	Global	0.169	$1.03x10^{7}$	0.0
	Individual	0.526	$1.69x10^{7}$	45.6
k.100.3	Global	0.202	$1.25x10^{9}$	4.4
	Individual	0.343	$1.44x10^{9}$	42.9
k.250.3	Global	0.103	$1.88x10^{10}$	0.0
	Individual	0.314	$3.20x10^{10}$	49.7
k.500.3	Global	0.095	$1.15x10^{11}$	0.0
	Individual	0.470	$2.29x10^{11}$	67.2
k.750.3	Global	0.035	$3.72x10^{11}$	0.0
	Individual	0.419	$1.23x10^{12}$	87.5
k.100.4	Global	0.084	$4.84x10^{12}$	0.0
	Individual	0.378	$7.60x10^{12}$	100.0
k.250.4	Global	0.071	$1.82x10^{14}$	0.0
	Individual	0.247	$3.74x10^{14}$	0.0
k.500.4	Global	0.039	$2.26x10^{15}$	0.0
	Individual	0.447	$6.94x10^{15}$	0.0
k.750.4	Global	0.050	$1.10x10^{16}$	0.0
	Individual	0.412	$4.10x10^{16}$	0.0

A comparison between individualized randomly shifted weights and a single global set of randomly shifting weights is presented in Table 11. In the majority of instances, individualized weights outperforms the single weight along all three metrics, with the exception of two anomalies. The smallest instance, k.100.2, appears to favour the use of a single weight, and the largest three instances, k.250-750.4, show no significant benefit to either constraint in terms of the univariate metric. The majority of presented results indicate that the extra computational effort to maintain individual weights is advantageous in terms of the quality of solutions. However, the anomalous portion of the results suggests that this design choice may be somewhat problem dependent.

5.9 Clustered Weights

The final experiment of this paper presents an example of a semi-intelligent form of objective weight setting. The intent here is to explore an example of a custom-made constraint for directing the search strategy is specific ways.

The Random Cluster constraint uses feedback from the search to direct the distribution of objective weights. Here each weight is either shifted randomly, as in the Random Shift constraint, or is moved towards the nearest weight set in the pareto-front.[8] Intuitively, by keeping the weights close to values that have been used to find some of the best-so-far solutions, one should expect a higher density of pareto

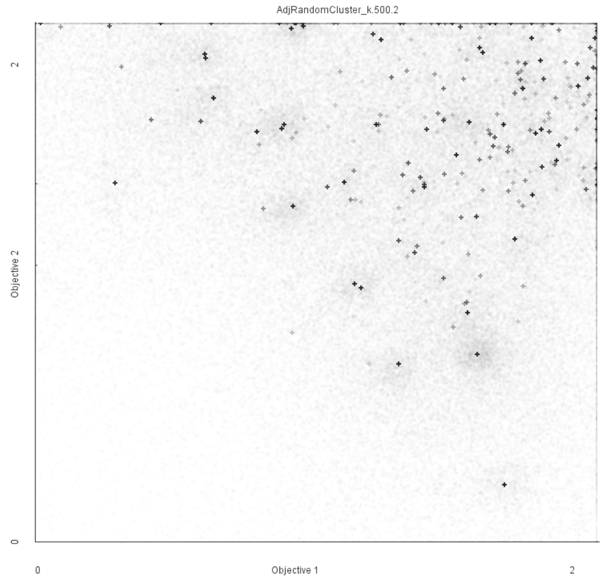

Figure 9: Objective weight sampling distribution using the random cluster constraints.

solutions. possibly at the cost of lower spread.

The implemented constraint produces a weight sampling distribution as seen in Figure 9. The fitness-space sampling distribution presented in Figure 10 appears slightly more compact than the equivalent Figure 8 for random shift. Considering the results in Table 12, the clustering has a consistent advantage in terms of the coverage measure, but a disadvantage in terms of the hypervolume. This suggests that clustering does in fact create higher density 'pockets' of pareto valued solutions. Inconsistent results in the univariate column suggests that the two constraint sets achieve comparable quality solutions overall.

6. CONCLUSION

The use of objective weights to balance search is common in many Multi-Objective Ant Colony Optimization implementations, however the impacts of its related design decisions are rarely considered. This paper examined the use of objective weights in terms of their impact on solution quality. It was found that diversity of objective weights amongst ants corresponded to a wider spread of solutions along the pareto front. Conversely, experiments using clustered weights demonstrated more compact 'pockets' of pareto solutions with limited spread. Common among many experimental results was the issue of scalability. It was found that the quality of solutions found using objective weights often improved with the complexity (in both size and dimension) of the benchmark problem instances. Also observed was a high degree of sensitivity in the correlation between changes in objective weight distribution and the pareto front solution distribution; this also exhibited greater influence as the problem complexity increased. Further findings indicated that dynamic shifts in a given ant's objective weight set were inconsequential so long as the instantaneous distribu-

[8] The original 'Cluster' constraint always moved the weights towards the nearest pareto set with some small amount of noise, but this was found to keep the weights too tightly clustered. The current Random Cluster constraint is a biased coin-flip deciding between using Random Shift or Cluster.

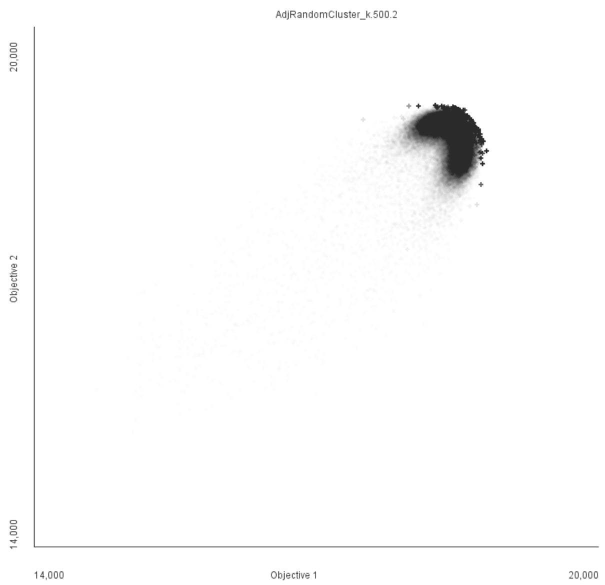

Figure 10: Fitness-space sampling distribution using clustered randomly shifting objective weights.

Table 12: Comparison of randomly shifting objective weights versus randomly clustering objective weights

Instance	Constraint	Coverage	Hypervol.	Univar.
k.100.2	Random	0.346	$1.00\text{x}10^5$	1.5
	Cluster	0.323	$1.13\text{x}10^5$	10.5
k.250.2	Random	0.220	$5.94\text{x}10^5$	0.0
	Cluster	0.562	$6.60\text{x}10^5$	40.2
k.500.2	Random	0.331	$1.98\text{x}10^6$	0.0
	Cluster	0.512	$1.75\text{x}10^6$	18.3
k.750.2	Random	0.278	$4.55\text{x}10^6$	0.0
	Cluster	0.538	$4.00\text{x}10^6$	11.2
k.100.3	Random	0.261	$6.62\text{x}10^8$	6.4
	Cluster	0.434	$4.14\text{x}10^8$	8.0
k.250.3	Random	0.315	$9.30\text{x}10^9$	28.8
	Cluster	0.401	$3.46\text{x}10^9$	0.28
k.500.3	Random	0.312	$6.08\text{x}10^{10}$	22.2
	Cluster	0.422	$4.34\text{x}10^{10}$	12.7
k.750.3	Random	0.198	$1.49\text{x}10^{11}$	0.0
	Cluster	0.598	$1.08\text{x}10^{11}$	35.7
k.100.4	Random	0.307	$2.48\text{x}10^{12}$	0.0
	Cluster	0.315	$1.65\text{x}10^{12}$	0.0
k.250.4	Random	0.250	$8.77\text{x}10^{13}$	100.0
	Cluster	0.430	$5.07\text{x}10^{13}$	0.0
k.500.4	Random	0.252	$1.09\text{x}10^{15}$	100.0
	Cluster	0.450	$6.40\text{x}10^{14}$	0.0
k.750.4	Random	0.158	$4.40\text{x}10^{15}$	0.0
	Cluster	0.568	$3.36\text{x}10^{15}$	0.0

tion of all objective weight sets in the population remained approximately the same. A comparison of a homogeneous population versus one with heterogeneous objective weights found that a variety of simultaneous objective biases present in the search at any given time results in higher quality solutions, despite the increased computational cost.

Future research should consider additional intelligently-designed and/or self-adaptive constraints, that are responsive to the state of the search itself, akin to the Random Cluster and Relative Quality constraints. Additional consideration should be given to the use of objective biasing weights related to the desirability matrices in addition to the pheromone matrices considered here. Further work into analysing the effects of other specific design choices, as described in the taxonomies, should also be pursued in order to generate concrete, highly performant instantiations of the MOACO metaheuristic.

7. REFERENCES

[1] I. Alaya, C. Solnon, and K. Ghedira. Ant colony optimization for multi-objective optimization problems. In *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence - Volume 01*, ICTAI '07, pages 450–457, Washington, DC, USA, 2007. IEEE Computer Society.

[2] J. Andersson. A survey of multiobjective optimization in engineering design. *Department of Mechanical Engineering, Linktjping University. Sweden.*

[3] D. Angus. Population-based ant colony optimisation for multi-objective function optimisation. In *Proceedings of the 3rd Australian conference on Progress in artificial life*, pages 232–244. Springer-Verlag, 2007.

[4] D. Angus. personal communication, 2011.

[5] D. Angus and C. Woodward. Multiple objective ant colony optimisation. *Swarm intelligence*, 3(1):69–85, 2009.

[6] B. Baran and M. Schaerer. A multiobjective ant colony system for vehicle routing problem with time windows. In *21 st IASTED International Multi-Conference on Applied Informatics*, pages 97–102, 2003.

[7] P. Cardoso, M. Jesus, and A. Márquez. Monaco-multi-objective network optimisation based on an aco. In *Proc. X Encuentros de Geometrıa Computacional*, Seville, Spain, 2003.

[8] K. Doerner, W. Gutjahr, R. Hartl, C. Strauss, and C. Stummer. Ant colony optimization in multiobjective portfolio selection. In *Proc. 4th Metaheuristics International Conference*, pages 243–248, 2001.

[9] K. F. Doerner, W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer. Pareto ant colony optimization with ilp preprocessing in multiobjective project portfolio selection. *European Journal of Operational Research*, 171(3):830–841, 2006.

[10] M. Dorigo and L. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. *Evolutionary Computation, IEEE Transactions on*, 1(1):53–66, 2002.

[11] M. Dorigo and T. Stützle. *Ant colony optimization*. Bradford Books. MIT Press, 2004.

[12] C. Fonseca and P. Fleming. On the performance assessment and comparison of stochastic multiobjective optimizers. *Parallel problem solving from nature*, pages 584–593, 1996.

[13] C. Garcia-Martinez, O. Cordón, and F. Herrera. A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria tsp. *European Journal of Operational Research*, 180(1):116–148, 2007.

[14] M. Guntsch. *Ant algorithms in stochastic and multi-criteria environments*. PhD thesis, Universität Fridericiana zu Karlsruhe, Germany, 2004.

[15] M. Guntsch and M. Middendorf. Solving multi-criteria optimization problems with population-based aco. In *Evolutionary Multi-Criterion Optimization*, pages 71–71. Springer, 2003.

[16] M. Hansen. *Metaheuristics for multiple objective combinatorial optimization*. PhD thesis, Institute of Mathematical Modelling, Technical University of Denmark, 1998.

[17] S. Iredi, D. Merkle, and M. Middendorf. Bi-criterion optimization with multi colony ant algorithms. In *Evolutionary Multi-Criterion Optimization*, pages 359–372. Springer, 2001.

[18] J. Knowles and D. Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary computation*, 8(2):149–172, 2000.

[19] M. López-Ibánez, L. Paquete, and T. Stützle. On the design of aco for the biobjective quadratic assignment problem. *Ant Colony, Optimization and Swarm Intelligence*, pages 436–459, 2004.

[20] M. López-Ibáñez and T. Stützle. An analysis of algorithmic components for multiobjective ant colony optimization: A case study on the biobjective tsp. *Artificial Evolution*, pages 134–145, 2010.

[21] T. Lust and J. Teghem. The multiobjective multidimensional knapsack problem: a survey and a new approach. *Arxiv preprint arXiv:1007.4063*, 2010.

[22] T. Lust and J. Teghem. The multiobjective traveling salesman problem: A survey and a new approach. *Advances in Multi-Objective Nature Inspired Computing*, pages 119–141, 2010.

[23] R. Marler and J. Arora. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395, 2004.

[24] T. Stützle and H. Hoos. MAX-MIN Ant system. *Future Generation Computer Systems*, 16(9):889–914, 2000.

[25] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms-a comparative case study. In *Parallel Problem Solving from Nature*, page 292. Springer, 1998.

[26] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *evolutionary computation, IEEE transactions on*, 3(4):257–271, 1999.