An Approach to Mitigating Unwanted Interactions between Search Operators in Multi-Objective Optimization

Chad M. Byers, Betty H.C. Cheng
BEACON Center for the Study of Evolution in Action
Department of Computer Science and Engineering
Michigan State University
East Lansing, Michigan, USA
{byerscha, chengb}@msu.edu

ABSTRACT

At run time, software systems often face a myriad of adverse environmental conditions and system failures that cannot be anticipated during the system's initial design phase. These uncertainties drive the need for dynamically adaptive systems that are capable of providing self-* properties (e.g., self-monitoring, self-adaptive, self-healing, etc.). Prescriptive techniques to manually preload these systems with a limited set of configurations often result in brittle, rigid designs that are unable to cope with environmental uncertainty. An alternative approach is to embed a search technique capable of exploring and *generating* optimal reconfigurations at run time. Increasingly, DAS applications are defined by multiple competing objectives (e.g., cost vs. performance) in which a *set* of valid solutions with a range of trade-offs are to be considered rather than a single optimal solution. While leveraging a multi-objective optimization technique, NSGA-II, to manage these competing objectives, hidden interactions were observed between search operators that prevented fair competition among solutions and restricted search from regions where valid optimal configurations existed. In this follow-on work, we demonstrate the role that niching can play in mitigating these unwanted interactions by explicitly creating favorable regions within the objective space where optimal solutions can equally compete and co-exist.

Categories and Subject Descriptors

I.2.8 [**Computing Methodologies**]: Machine Learning—*Machine Learning Approaches, Bio-inspired approaches, Genetic Algorithms*

Keywords

Decision making; Genetic Algorithms; Multi-objective optimization; Multiple solutions / Niching; Reference point-based niching; NSGA-III; Remote Data Mirroring

GECCO '15, July 11 - 15, 2015, Madrid, Spain

ISBN 978-1-4503-3472-3/15/07...$15.00

DOI: http://dx.doi.org/10.1145/2739480.2754698

1. INTRODUCTION

Intended to address the challenges posed by adverse environmental conditions and varying user requirements, dynamically adaptive systems (DASs) [18, 21] must reliably self-reconfigure at *run time* to avoid critical data and financial losses [17]. Emerging applications include the management of smart energy grids, traffic, and telecommunication systems that must cope with a myriad of system and environmental uncertainties (ex. node failures, link failures, etc.). Manual approaches to preload these systems with solutions are *prescriptive* in nature and often lead to brittle, rigid designs that are unable to scale or adapt to unanticipated conditions. An alternative approach is to embed a search process capable of exploring and *generating* new reconfigurations as the environment or high-level objectives/goals change. Evolutionary search techniques, such as genetic algorithms (GAs) [13, 15], provide one approach for DAS reconfiguration by harnessing biological principles of evolution to explore a *suite* of candidate solutions that often vary in their size, complexity, and the trade-offs made across the problem's dimensions. Previously [6], the underlying solution encoding was observed to have unintended interactions with specific evolutionary search operators, prohibiting search from returning solutions in valid regions of the objective space. This paper explores the use of niching to mitigate the negative effects of these unwanted interactions on solution set diversity.

In order to increase access to solutions throughout the objective space, a series of GA-based tools have been progressively developed, each of which required different levels of domain expertise: Weighted Plato [23], Targeting Plato [5], and NSGA-II Plato [6] that support dynamic reconfiguration for an industrial data mirroring (RDM) application. While the evolved solutions of these approaches were better than solutions generated manually, further inspection revealed that solutions were ill-fit for their intended environment, regions of the objective space were unreachable, or an exhaustive, trial-and-error-process would be necessary to yield desired solutions. Recently [6], hidden interactions were observed between a variable-length genome design and the diversity-preserving search operator of a popular multi-objective evolutionary algorithm (MOEA), NSGA-II[1], that significantly restricted search coverage in the objective space. This bias resulted from the violation of a key assumption that solutions' objective values are equally affected by mutation. More specifically, a variable-length genome design causes solutions with fewer elements to experience a greater change to their objective values when compared to more complex solutions. Since the diversity-preserving operator of NSGA-II prioritizes solutions located in sparsely populated regions, solutions with

[1]Non-dominated Sorting Genetic Algorithm [12]

fewer elements are more readily accepted due to their increased likelihood of escaping crowded regions. As a result, candidate solutions of varying size are unable to *globally* compete with one another in the same population.

To support a diverse set of optimal solutions where solution complexity is a key variable being explored, we propose that **niching** [25] be leveraged to mitigate unwanted, hidden interactions between search operators. Niching, in this work, refers to the process by which favorable regions are explicitly created and maintained in the objective space thereby partitioning a population into disjoint subpopulations. Niching enables fair competition during search since members of the same subpopulation, or niche, share similar objective values (e.g., complexity) and compete *locally* with one another. The ability to explicitly define niches within the objective space enables techniques that harness niching to drive search towards regions difficult to reach for other techniques.

We use the reference point-based niching operator of NSGA-III [10] to generate a diverse set of target configurations for an industrial RDM application where the number of solution elements (e.g., nodes, links, etc.) evolves freely over time. While this niching operator was intended to counteract the Curse of Dimensionality Problem [2] for high-dimensional applications, we demonstrate, in this work, its ability to successfully mitigate the hidden search interactions/bias present in NSGA-II and return diverse sets of Pareto-optimal configurations throughout the objective space. We compare NSGA-III's performance to previous Plato approaches in several key areas including overall search coverage and efficiency.

The remainder of this paper is organized as follows. In Sections 2 and 3, we provide background regarding RDM systems and an overview of preliminary investigations with the Plato approaches as well as their shortcomings. Next, Section 4 highlights three high-level properties necessary for search techniques used in DASs. We introduce NSGA-III's niching operator in Section 5 and discuss the key differences between NSGA-II and NSGA-III. Section 6 details our experimental design and results assessing NSGA-III's ability to provide these necessary high-level properties. Lastly, Section 7 overviews related work and we summarize our key findings and outline future directions for this work in Section 8.

2. BACKGROUND

In this section, we describe the application domain for a DAS, namely, remote data mirroring systems as well as the key elements and challenges in their design.

2.1 Remote Data Mirroring (RDM)

Remote data mirroring (RDM) systems [17] are widely deployed to offer file synchronization and reliability for critical data items distributed across a network in the presence of site/link failures [17]. In order to accomplish these objectives, RDM systems *copy* critical data residing at primary sites and *remotely store* (mirror) this data across secondary sites within the network. The design of an RDM solution, referred to as an **overlay network** [1], centers around two critical decisions: (1) the subset of network links to include from the underlying base network and (2) the type of RDM networking protocol utilized on each active link.

Two networking protocol types, *synchronous* and *asynchronous*, exist to balance the trade-offs associated with improving overall network bandwidth (performance) while mitigating the risk for greater potential data loss (reliability). A *synchronous* protocol ensures that each write operation of a critical data item is successfully applied and stored at the secondary site before subsequent write operations can proceed at the primary site. In contrast, *asynchronous* protocols *coalesce* write operations at the primary site over a specified length of time. After this length of time has elapsed, write operations are sent in batch form over the network to secondary sites and applied atomically. In Table 1, we present the elapsed time between each batch, the amount of potential data at risk (in GB), as well as the percentage of consumed network bandwidth for both synchronous (P1) and asynchronous (P2-P7) protocols.

In addition to its inherent DAS properties (e.g., self-* properties), the RDM problem encompasses a vast solution space where the set of all possible network designs is too large to manually enumerate. For a fully-connected network, the RDM problem's order of complexity comprises $2^{n(n-1)/2}$ network constructions, where n is the number of mirror sites.

Protocol Type	ID	Interval Length	Data at Risk (GB)	Bandwidth
Synchronous	P1	0 minutes	0.0	1.0
Asynchronous	P2	1 minute	0.35	0.9098
	P3	5 minutes	0.6989	0.8623
	P4	1 hour	1.7782	0.7271
	P5	4 hours	2.3802	0.5732
	P6	12 hours	2.8573	0.4380
	P7	24 hours	3.1584	0.3967

Table 1: Properties of RDM networking protocols [17].

3. PRELIMINARY INVESTIGATIONS

As part of an industrial collaboration, a series of genetic-algorithm based tools have been progressively developed in an effort to improve access to optimal solutions across the full range of dimensions in a RDM application.

3.1 Solution Encoding and Problem Objectives

For each of the three previous Plato approaches discussed in this paper (i.e. Weighted Plato, Targeting Plato, NSGA-II Plato), a genetic algorithm is integrated into the decision-making process of a RDM system in order to dynamically evolve reconfiguration plans, or overlay networks, that balance various objectives. An overlay network is internally represented as a vector containing N total elements. Each element maps to a specific connection (link) in the underlying base network and stores (1) a boolean flag for whether the connection is in use by the solution and (2) the specific RDM protocol used on the active connection. This solution encoding is subject to random mutation and crossover to generate new overlay network constructions.

During optimization of an RDM system, three orthogonal objectives (i.e. dimensions) are targeted: Cost (f_{cost}), Performance (f_{perf}), and Reliability (f_{reliab}). To quantify the optimization level that each network was able to achieve across these dimensions, we used the aggregate formulas given in Equations (1)-(6) that were derived from studies for optimizing data recovery systems [17]. In these equations, the vector ($\mathbf{x}$) corresponds to an evolved overlay network solution containing N total elements. For each vector element ($\mathbf{x}_i$) that maps to a unique link in the base network, the $\mathbf{x}_i^{flag}$ property indicates whether the link is active (1) or inactive (0) and $\mathbf{x}_i^{risk}$ and $\mathbf{x}_i^{bandwidth}$ indicate the data at risk and bandwidth consumed by the RDM protocol in use on the link, respectively. To calculate an overlay network's cost, the operational expense for activating an underlying network link is denoted as C_i. Similarly, the properties of particular RDM protocols, such as the amount of data at risk (ex. $P2^{risk}$), refer to those values given in Table 1. To remove the bias for search to favor dimensions with larger ranges of values, each objective function has been normalized between 0.0 and 1.0.

$$f_{cost}(\mathbf{x}) = \frac{\sum_{i=0}^{N} C_i x_i^{flag}}{\sum_{i=0}^{N} C_i} \quad (1)$$

$$f_{perf}(\mathbf{x}) = \frac{f_{efficiency}(\mathbf{x}) - P1^{bandwidth}}{P1^{bandwidth} - P7^{bandwidth}} \quad (2)$$

$$f_{efficiency}(\mathbf{x}) = \frac{\sum_{i=0}^{N} x_i^{bandwidth} x_i^{flag}}{\sum_{i=0}^{N} P1^{bandwidth} x_i^{flag}} \quad (3)$$

$$f_{reliab}(\mathbf{x}) = 0.5 \times f_{reliab1}(\mathbf{x}) + 0.5 \times f_{reliab2}(\mathbf{x}) \quad (4)$$

$$f_{reliab1}(\mathbf{x}) = 1.0 - \frac{\sum_{i=0}^{N} x_i^{flag}}{N} \quad (5)$$

$$f_{reliab2}(\mathbf{x}) = \frac{\sum_{i=0}^{N} x_i^{risk} x_i^{flag}}{\sum_{i=0}^{N} P7^{risk} x_i^{flag}} \quad (6)$$

For each of the preliminary investigations discussed in this section, the goal was to evolve overlay network solutions for a *fully-connected* RDM network containing 26 data mirrors. Each experimental run contained a population of 500 candidate solutions and employed tournament selection ($k = 5$), two-point crossover, and a 5% mutation rate for a total of 1,000 generations. To provide adequate statistical significance, a unique random seed was assigned to 30 replicate runs for each weight/target combination.

3.2 Weighted Plato

To guide evolutionary search towards solutions aligned with a user's high-level goals, a linear-weighted sum, shown in Equation (7), was used in Weighted Plato to prioritize problem objectives. During search, evolved network solutions whose characteristics maximized this linear-weighted sum, or fitness function, were considered to be aligned with the user's goals and selected by the genetic algorithm.

$$\begin{aligned} \text{Maximize} \quad & \alpha_{cost}(1.0 - f_{cost}) + \\ & \alpha_{perf}(1.0 - f_{perf}) + \\ & \alpha_{reliab}(1.0 - f_{reliab}) \end{aligned} \quad (7)$$

To evaluate how the relative prioritization of problem dimensions guided evolutionary search in the objective space [5], a diverse set of unique coefficient combinations were systematically generated by using an initial coefficient value of 0.0, an increment size of 0.10, and a requirement that the weighting coefficient values sum to 1.0. For each of these 66 unique coefficient combinations, the Weighted Plato reconfiguration tool was applied to dynamically generate an evolved overlay network optimized for the specific prioritization of problem objectives.

Experimental Results.

The Weighted Plato plot in Figure 1a shows the *combined* results from all 66 combinations where each three dimensional point corresponds to a returned solution's network measures. Although valid solutions were generated, three critical weaknesses were exposed. First, the solution surface where Pareto-optimal solutions reside was found to be non-convex causing a large number of valid solutions to be undetectable using a linear-weighted approach [7]. Second, after inspecting the qualities of returned solutions, we often found that solutions were optimized along a *single* dimension and misaligned with the user's weights and therefore, ill-fit for their intended environment. Third, linear-weighted sum approaches sacrifice exploration of the objective space in order to optimize a fitness function, and, as a result, a large number of coefficient combinations are necessary to achieve good search coverage.

3.3 Targeting Plato

To address the aforementioned challenges when using a linear-weighted approach, Targeting Plato [5] was developed, where a user specified the ideal *target values* for solutions to achieve along each objective. Candidate solutions in Targeting Plato were rewarded for their proximity to the ideal solution's target values as shown in Equation (8).

As before, we systematically generated various target configurations by starting each target measure at 0.0 with an increment size of 0.10 until the maximum boundary of 1.0 was reached within each dimension. We applied the Targeting Plato reconfiguration tool to generate an overlay network and plotted the *combined* solution set across all 1331 unique combinations in Figure 1b.

$$\begin{aligned} \text{Minimize} \quad & |\tau_{cost} - (1.0 - f_{cost})| + \\ & |\tau_{perf} - (1.0 - f_{perf})| + \\ & |\tau_{reliab} - (1.0 - f_{reliab})| \end{aligned} \quad (8)$$

Experimental Results.

We observed that Targeting Plato was not susceptible to the Pareto surface's non-convexity as solutions were successfully returned in interior regions where Weighted Plato was unable to previously reach, as shown in Figure 1b. To ensure that Pareto-optimal solutions were targeted during search, this approach requires that the target points be placed either on or beneath the Pareto surface. However, *apriori* knowledge of where the Pareto surface lies is rarely available causing a domain expert to rely on a trial-and-error approach with various target combinations. While Targeting Plato provided a more intuitive specification method for domain experts, this approach also collapsed the problem dimensions into a *single* objective function during optimization, similar to Weighted Plato. As such, the returned solution sets of these approaches often lacked diversity in their trade-offs, thus requiring costly iterations of search to be performed.

3.4 NSGA-II Plato

Two recurring observations with previous Plato-based studies were that (1) the objectives were often competing against, or in some cases negating, the effects of other objectives, or (2) complete knowledge of the fitness landscape was necessary to avoid suboptimal solutions. Despite only three objectives being present in the RDM application, multi-objective evolutionary algorithms (MOEAs) were explored next for their ability to manage competing objectives. As shown in Equation (9), each objective is optimized simultaneously in order to return a diverse set of Pareto-optimal solutions where each solution is optimal with respect to its associated trade-offs. These algorithms also do not require domain experts to specify desired solution qualities or to prioritize objectives to determine where solution trade-offs occur. Recently [6], a commonly-used MOEA named NSGA-II [12] was leveraged to mitigate the drawbacks of both Weighted Plato and Targeting Plato due to its incorporation of two operators: (1) non-dominated sorting and (2) a crowding-distance operator. Non-dominated sorting ensures that solutions approach true Pareto optimality by prioritizing solutions whose objective measures dominate the objective measures of other solutions in the population. To diversify the range of trade-offs found in the returned solution set, NSGA-II's crowding distance operator prioritizes solutions located in less crowded (i.e. novel) regions of the objective space.

$$\text{Minimize} \quad (f_{cost}, f_{perf}, f_{reliab}) \quad (9)$$

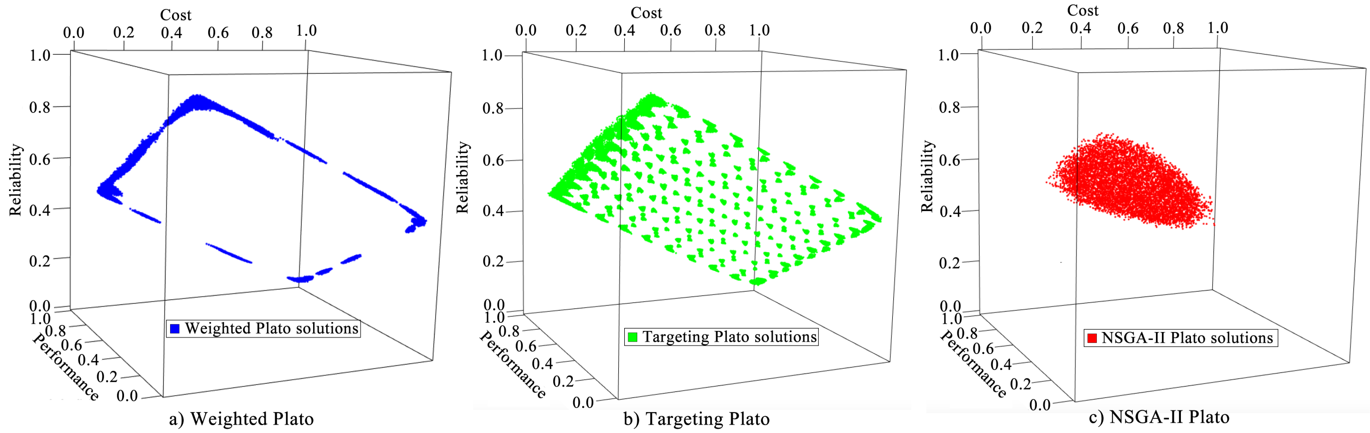

Figure 1: Returned solutions plotted within three-dimensional (Cost, Performance, and Reliability) objective space.

Experimental Results.

NSGA-II Plato was observed to overcome several shortcomings of Weighted Plato and Targeting Plato by returning solutions in non-convex regions without the requirement of domain-specific inputs (e.g., coefficients or target values) or complete knowledge of the Pareto surface. However, *hidden interactions* were discovered that restricted NSGA-II from regions of the objective space where valid, Pareto-optimal solutions were known to exist. Through an empirical study, these hidden search interactions were demonstrated to arise when solution objectives were unequally affected by mutation. In the context of the RDM application, solution objectives were unequally affected by mutation due to a variable-length genome design used to freely evolve overlay networks of various sizes and complexities. As shown in Figure 1c, NSGA-II Plato's search was biased towards regions associated with low Cost values as these solutions contain few network links and thus their objective values experience greater changes during mutation. Since NSGA-II's crowding-distance operator prioritizes solutions located in sparsely-populated regions of the objective space, solutions whose objective values experience greater change are favored since they are less likely to become more crowded over time.

4. DESIRED SEARCH PROPERTIES

Given the observed weaknesses of previous Plato approaches, we highlight three high-level abilities needed for search techniques used in the dynamic reconfiguration of DAS systems: (1) the ability to return a *diverse* solution set, (2) the ability to return *non-dominated, Pareto-optimal* solutions, and (3) the ability to *manage non-uniform mutational effects* to solution objectives. For each of these abilities, we provide a working definition, describe its importance for DAS reconfiguration, and summarize whether the ability was supported in previous Plato approaches (Table 2).

4.1 Diverse Solution Set

Solution set diversity, or "spread," represents the degree of coverage a search technique is able to achieve in the objective space. Maintaining a diverse population preserves a variety of open, parallel search avenues to explore, thus lowering the barrier of probabilistically discovering a global optima, if multiple paths to such optima exist. While a diverse population may slow convergence on newly discovered optima due to its spread, this property also reduces the likelihood of search becoming trapped in local optima. For applications, such as RDM, that require a DAS to respond to a variety of environmental and system uncertainties, a highly diverse solution set offers (1) a range of trade-offs for both domain users and automated tools to consider as well as (2) a method to indirectly probe the range of achievable objective values given current conditions.

In both the original Weighted Plato and Targeting Plato, the quality of an evolved solution was collapsed into a single fitness value, and, as a result, search often converged upon *one* particular solution type that optimized this value. The returned solution sets of these approaches often lacked diversity, and good coverage could only be achieved by exhaustively varying search parameters over many search iterations. In addition, non-convex Pareto surfaces prevent Weighted Plato from returning a large number of solutions. While the crowding-distance operator of NSGA-II Plato is normally capable of driving search towards sparsely-populated areas of the objective space, hidden interactions with solutions whose objective values are unequally affected by mutation were observed to disrupt this mechanism. Therefore, NSGA-II only supports solution set diversity when these interactions are not present in the optimization problem.

4.2 Non-Dominated, Pareto-Optimal Solutions

For applications defined by multiple orthogonal (conflicting) objectives, a solution is *Pareto-optimal* if no single objective can be improved further without sacrificing its quality along another objective. While Pareto optimality can be assessed for a solution in isolation, non-domination is assessed relative to other members of the *aggregate* solution set, or population. A solution B is said to "dominate" another solution C when solution B is no worse than solution C within each respective objective and solution B improves upon at least one of solution C's objective values. The ability to return an increasing majority of non-dominated solutions is a desirable search property since these solutions are irreplaceable by other population members thus helping to minimize search "waste" (i.e., dominated solutions) and improve search efficiency. However, since minor changes may easily produce new, non-dominated solutions, this ability alone does not guarantee a broad range of trade-offs, thus underscoring the importance of solution set diversity.

Linear-weighted sum techniques, such as Weighted Plato, can be used to seek Pareto-optimal solutions, however, non-convex regions of the objective space remain undetectable and, as demonstrated, returned solutions may be misaligned with the user's goals and therefore ill-fit for their intended environment. While Targeting Plato overcame these weaknesses, *apriori* knowledge of the Pareto surface's location is necessary to avoid returning suboptimal

	Desired Search Property		
	Seeks Diverse Solution Set (Coverage)	**Seeks Non-dominated (Pareto-optimal) solutions**	**Manage non-uniform mutational effects on solution objectives**
Weighted Plato (Weighted Sum)	No	**Yes**	N/A
Targeting Plato (Ideal Solution)	No	**Yes**	N/A
NSGA-II Plato (Crowding Distance)	**Yes**	**Yes**	No

Table 2: The desired search properties supported by previous Plato approaches.

solutions causing a domain expert to often rely on a trial-and-error approach using various target combinations. By constantly prioritizing solutions that improve upon the objective values of other population members, NSGA-II Plato is able to seek and return non-dominated, Pareto optimal solutions, however, these solutions were not representative of the overall objective space.

4.3 Manage Non-Uniform Mutational Effects

Finally, in order to handle combinatorial optimization problems, such as our RDM application, we seek evolutionary search techniques that can successfully mitigate the non-uniform effects of mutation on solution objectives as a direct result of hidden interactions recently observed [6]. In this empirical study, a variable-length genome caused solutions' objective values to be disproportionately affected by mutation and as a result, solutions with fewer elements experienced greater changes to their objective values during mutation thus conferring an advantage for escaping crowded regions. These hidden interactions between NSGA-II's crowding-distance operator and mutation prevented fair competition among solutions and biased search towards regions containing smaller solutions. In various application domains, the number of solution elements is often a free variable evolved to explore different designs and their associated trade-offs. Therefore, properly managing this interaction is critical for search techniques used for DAS. Moreover, a variable-length genome design is but one specific example by which objective values may vary disproportionately during search.

Both Weighted Plato and Targeting Plato approaches do not contain a mechanism, such as the crowding-distance operator, that explicitly rewards solutions based on their location relative to one another. As a result, these approaches are not susceptible to non-uniform mutational effects.

5. LEVERAGING NICHING

In biology, an ecological niche [22] often refers to a specific role, function, or area that an organism inhabits within an ecosystem as a result of the distribution of resources, predators, and/or competitors. When space within a population of competing individuals is limited, evolution via natural selection often drives organisms to enter unique niches in order to minimize competition thus allowing a more diverse overall population to co-exist. Similarly, in evolutionary search, a niching operator *creates* regions where individuals can locally compete by segmenting the population into distinct subpopulations and is useful when *simultaneously* targeting multiple optima and/or when solution set diversity is difficult to achieve due to convergence.

5.1 NSGA-III

In many-objective (four or more objectives) optimization problems, two critical issues arise: (1) the search space volume increases exponentially, due to the Curse of Dimensionality problem [2], making it difficult to effectively explore new regions, and (2) the majority of the population's open slots become occupied with non-dominated solutions causing evolutionary search to slow down dramatically [10]. To address these issues, Deb and Jain recently developed a new MOEA named NSGA-III [10] that incorporated the concept of **niching** during search by replacing NSGA-II's crowding-distance operator with a *reference point-based niching* operator.

We propose a new alternative role for NSGA-III's niching operator, namely, to manage non-uniform mutational changes to solution objectives. In applications where variable-length genomes are used, solutions cannot globally compete with one another due to inequity created by differing solution sizes. Niching promotes fair competition among solutions by ensuring that solutions *locally* compete with others based on similar objective values thereby building equity between solutions back into the optimization algorithm. In addition, the ability to explicitly define niche locations in the objective space allows NSGA-III to produce a selective pressure for search to enter regions previously difficult to reach by NSGA-II.

To understand the key differences between the diversity-preserving mechanisms of NSGA-II (*crowding-distance*) and NSGA-III (*niching*), we describe the distinct process used by each operator to select solutions for subsequent iterations of search. We begin, however, with an overview of shared core functionality found in both NSGA-II and NSGA-III approaches. At the start of each generation, a parent population (**P**) containing N individuals is used to generate an offspring population (**Q**) of equal size using standard genetic operators such as crossover and mutation. Next, these two populations are merged into one combined population (**R**) where all $2N$ solutions are sorted into a series of disjoint, non-dominated sets called *fronts*. Solutions are iteratively placed into successive fronts (ex. F_1, F_2, etc.) if they are non-dominated with respect to remaining, unsorted solutions. To select for truly Pareto-optimal solutions, fronts are sequentially added based on their non-domination level until the population capacity, N, is matched or exceeded. If the population capacity is matched upon adding the last front (F_L), no further operations are necessary and the next generation begins again using this newly generated population. If adding the last front (F_L) causes the population capacity to be exceeded since only K slots remain, then additional steps must be taken to determine which solutions from F_L will fill these remaining unoccupied slots. The two distinct diversity-preserving mechanisms, crowding-distance and reference point-based niching, used to make this decision are discussed below.

Crowding Distance Operator (NSGA-II).

In the NSGA-II implementation used in NSGA-II Plato [6], a *crowding-distance* operator calculated each solution's nearest neighbor distance based on all of the problem objectives. Since solutions that maximized this distance measure were more likely to be located in novel regions of the objective space, the top K solutions of F_L filled the remaining slots in the new population. This diversity-preserving mechanism is intended to expand overall coverage by driving search towards sparsely-populated regions of the objective space. However, this mechanism can be disrupted when solution objectives are disproportionately affected by mutation.

Reference-Point Based Niching Operator (NSGA-III). In NSGA-III, the *reference point-based niching* operator starts with a set of reference points, or niches, that either (1) have been generated in a structured manner, such as being uniformly distributed, or (2) pre-determined and supplied by the user. Next, solutions from previously added fronts (i.e., $F_1 \dots F_{L-1}$) are *associated* to a unique niche. To associate a solution to a specific niche, a reference line is projected from the origin through each niche's reference point, and the perpendicular distance between a solution and each reference line is calculated. Afterwards, each solution is associated with the niche whose reference line has the closest perpendicular distance. To fill the remaining K slots in the next generation's population, the niche with the least associated solutions (excluding niches with no associated solutions) is chosen and the solution in F_L with the closest perpendicular distance to this niche's projected reference line is added to the population. After this solution has been associated with the niche, this selection process repeats until all remaining K slots are filled.

6. EXPERIMENTAL DESIGN AND RESULTS

We assess whether niching can provide the three high-level desired properties for DAS by using NSGA-III for the dynamic reconfiguration of an industrial RDM application. In this application, NSGA-III must evolve overlay network solutions for a *fully-connected* RDM network containing 26 data mirrors, causing the optimization problem to be non-trivial based on the large number of unique network constructions possible (i.e., 7×2^{325}).

Each evolutionary run contained a population of 500 candidate solutions that employed tournament selection ($k = 5$), two-point crossover, and a 5% mutation rate for a total of 1,000 generations, equating to roughly 4 minutes of wall-clock time. To provide adequate statistical significance, we used 30 replicate runs with a unique random seed assigned to each run. We used Das and Dennis's [8] systematic approach to generate a uniformly distributed grid of reference points in the objective space on a normalized hyperplane in $N-1$ dimensions. The granularity of this reference grid's resolution can be specified by the domain expert based on the number of intervals (p) along each of the hyperplane's edges. The total number of reference points (H) for an M-objective problem is given by Equation (10). In our RDM application where solutions were to be optimized along three objectives ($M = 3$), each of the 2D hyperplane's edges was divided into 20 intervals ($p = 20$), producing a total of 231 unique uniformly-distributed reference points.

$$H = \binom{M+p-1}{p} \qquad (10)$$

Diverse Solutions and Mitigating Non-Uniform Effects. To assess solution set diversity due to niching, we visually inspected the search coverage for each replicate run using NSGA-III Plato. In order to demonstrate that each run consistently achieved the same level of coverage, solutions from an individual run were assigned the *same*, unique color along a red-blue gradient (i.e., $Run_1 = Red \dots Run_{30} = Blue$). Using niching, we observed that a significantly large region of the Pareto surface was consistently found and returned by each experimental run when compared to the achieved coverage of previous Plato approaches (Figure 1). As such, the even distribution of each run across the full objective space has yielded an overall purple coloring of the combined results. More importantly, solutions of varying sizes and complexities were evolved, maintained, and returned. This result demonstrates that niching was able to overcome and mitigate the hidden search bias caused by non-uniform changes to solution objectives, a critical property for maximizing the number of different DAS designs as well as their associated trade-offs.

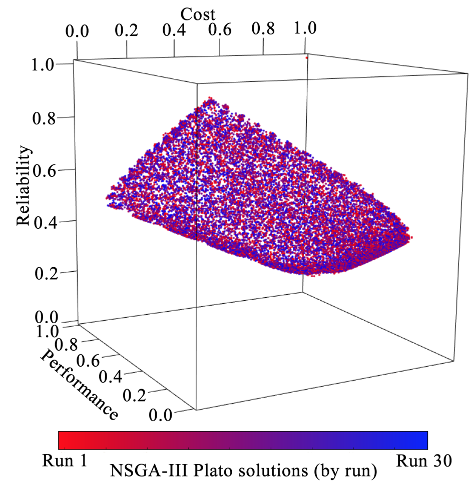

Figure 2: The returned solutions for NSGA-III Plato plotted within three-dimensional objective space.

To quantify and compare the search volume for each of the Plato approaches, we used the *hypervolume indicator* [27] to measure the volume dominated by Pareto-optimal solutions in the final population. The percentage of the objective space's total hypervolume that is dominated by a *single* run across the different Plato approaches is summarized in Figure 3.

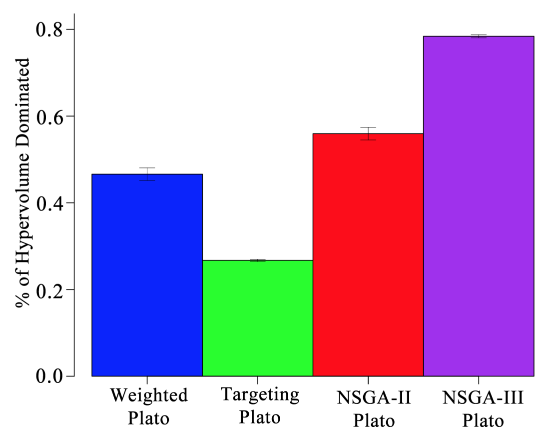

Figure 3: Average percentage of the hypervolume dominated by each Plato approach's solutions

By leveraging niching, NSGA-III achieved a significant increase ($p \ll 0.01$) in average search coverage that was 1.6-1.7 times greater than the average Weighted Plato run, 2.9-3.0 times greater than the average Targeting Plato run, and 1.4 times greater than the average NSGA-II Plato run. However, due to the non-convex Pareto surface, Weighted Plato could not achieve equivalent search coverage, even with multiple runs, since the interior of this surface is unreachable by linear-weighted approaches. Using numerous target value combinations, Targeting Plato could achieve comparable search coverage, however, this is computationally expensive and requires *complete* knowledge of the Pareto surface's location to ensure sub-optimal solutions are not returned. Similar to Weighted Plato, the NSGA-II Plato approach could not achieve equivalent search coverage due to the non-uniform changes to solution objectives during mutation that restricts search to a limited region of the Pareto surface.

7. RELATED WORK

The hidden search bias caused by non-uniform changes to solutions' objective values was only discovered recently [6] and therefore, to the best of the authors' knowledge, the use of reference point-based niching to mitigate this issue has not been previously presented in the literature. The key factor that enabled reference point-based niching to overcome this hidden search bias was that niches were *explicitly defined* and maintained during search. Alternative niching strategies [25] often rely on niches to *dynamically emerge* in the objective space based on solution properties.

In **sharing** and **clearing** strategies [14, 16, 24], a resource (e.g., objective value or fitness) is apportioned to individuals within a user-defined niche radius (threshold) in either the solution or design space. In these strategies, niches emerge as solutions are driven towards sparsely-populated regions to avoid sharing and competing for their niche's resource with more population members. However, these strategies may be biased against solutions whose objective values experience little variation as these solutions more rapidly acquire niche members. **Clustering** operators [25], such as k-means, can be used in place of a niche radius for allocating individuals into k niches based on their proximity to a centroid, however, offline analysis of this approach demonstrated that this approach was unsuccessful at mitigating the search bias.

In **crowding** strategies [9, 12], competition takes place between parents and offspring for inclusion in the next generation's population, where offspring crowd out the most similar individual from a randomly selected subset of user-specified size, often referred to as the crowding factor. In these strategies, niches emerge as solutions are driven away from other members to lower the probability of being selected for replacement. Using NSGA-II, we observed [6] that crowding strategies remain susceptible to search bias as parents whose objectives experience high variation are likely to be more dissimilar to their offspring and therefore may be prioritized during search.

Similarly, the **ϵ-dominance** [11] strategy expands the range of objective values considered to be *dominated* by a particular solution by adding a small value (ϵ) to each objective measure. Niches emerge as solutions are driven away from other members to lower the probability of being placed into a dominated front. However, this strategy is vulnerable to similar hidden interactions since it requires niches to be dynamically created by solutions as opposed to explicitly being defined. Once a solution becomes truly Pareto-optimal, neighboring solutions must be able to mutate outside the range of objective values considered to be dominated which may be difficult in regions where these values experience less variation, thus biasing search against these regions.

Reference grid niching strategies typically divide the objective space into subspaces either through an explicit distribution of niches (e.g., NSGA-III) or by allowing niches to emerge from solution properties. The Pareto Archived Evolutionary Strategy (PAES) [19] performs the latter by recursively bisecting the range of current objective values to determine whether to archive a solution. One potential shortcoming of this strategy, however, is that the degree to which a solution is crowded affects the archival of solutions. Since archived solutions whose objective values experience less variation become crowded more rapidly, these solutions may be frequently selected by PAES for replacement and biased against. In contrast, reference grid strategies that explicitly define niches, such as NSGA-III, are not susceptible to the level of crowding within a niche, and therefore the niches are preserved by keeping at least one associated member in subsequent generations of search.

8. CONCLUSIONS

The experiments in this paper have demonstrated the value of *reference point-based niching* for applications defined by two key features: (1) solution set diversity is a priority and (2) objective values of solutions are unequally affected by mutation. Multi-objective genetic algorithms, such as NSGA-II, often seek to return a *diverse* set of Pareto-optimal solutions and may incorporate search operators (e.g., crowding-distance) that are responsive to the distances that solutions mutate in the objective space. When solutions' objective values are disproportionately affected by mutation, however, these operators are biased towards solutions whose objective values experience greater changes due to their ability to escape into less crowded, novel regions of the objective space.

Reference point-based niching mitigates this issue by performing two key functions. First, niches are *explicitly* created and maintained uniformly throughout the objective space in regions difficult for search to reach. The solutions in these regions may either be too difficult to construct by random mutation alone or biased against due to smaller changes to their objective values. Niching produces a strong selective pressure for solutions to find sparsely populated niches in order to minimize competition and increase their chances of survival into the next generation. As a result, solutions are constantly driven to equally spread across the objective space resulting in the significant observed increase in search coverage (i.e. diversity) when compared with previous approaches. Second, reference point-based niching ensures that only solutions with similar properties (i.e. solution size or Cost) are localized within the same niche promoting fair competition among solutions.

Although the variable-length genome design in the RDM application caused non-uniform changes to solution objectives, this scenario is only one of many different ways that this hidden search bias may manifest itself in an application. For example, if the mapping between solution elements and problem objectives changes over time, then solutions whose objectives are mapped to fewer elements will experience greater changes to their objectives, and, as a result, these solutions may be preferred during search. Alternatively, if an objective's granularity is non-uniform, then a search bias may exist for solutions in more coarse-grained regions of the objective that experience more variation. Moreover, if the granularities between objectives are different, then solutions able to optimize along the coarser objective more easily may be prioritized. Objective function granularity has been recently highlighted [20] as an important future research area for its effects on search dynamics and performance. These issues are exacerbated further in dynamic optimization problems (DOPs) [4] as both the Pareto surface and granularity of objectives change over time thus underscoring the importance of niching.

Our results provide a key insight into niching features useful for overcoming this hidden bias to aid researchers exploring multi-objective optimization in domains sharing similar qualities. In general, our future work aims to explore problem and solution characteristics that cause evolutionary search techniques to be either well-suited or counterproductive to multi-objective optimization. Future directions include (1) exploring adaptive methods for distributing reference points, (2) designing a formal benchmark problem for studying both spatial and temporal changes to objective granularity, and (3) evaluating alternative niching strategies' performance on this problem such as ϵ-dominance as well as hypervolume-based approaches such as IBEA [26] and SMS-EMOEA [3].

9. ACKNOWLEDGMENTS

We gratefully acknowledge Professor Kalyanmoy Deb for his help introducing us to NSGA-III and his feedback on this work. This work has been supported in part by NSF grants IIP-0700329, CCF-0820220, DBI-0939454, CNS-0854931, CNS-0915855. Any opinions, findings, or conclusions expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or other sponsors.

10. REFERENCES

[1] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient overlay networks. *SIGOPS Oper. Syst. Rev.*, 35(5):131–145, Oct. 2001.

[2] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1st edition, 1957.

[3] N. Beume, B. Naujoks, and M. Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, September 2007.

[4] J. Branke, T. Kaubler, C. Schmidt, and H. Schmeck. A multi-population approach to dynamic optimization problems. In *In Adaptive Computing in Design and Manufacturing*, pages 299–308. Springer, 2000.

[5] C. Byers and B. H. C. Cheng. Mitigating uncertainty within the dimensions of a remote data mirroring problem. Technical Report MSU-CSE-14-10, Computer Science and Engineering, Michigan State University, East Lansing, Michigan, September 2014.

[6] C. Byers, B. H. C. Cheng, and K. Deb. Unwanted feature interactions between the problem and search operators in evolutionary multi-objective optimization. In *Proceedings of the 8th International Conference on Evolutionary Multi-Criterion Optimization*. Springer-Verlag, 2015 (full paper - to appear).

[7] I. Das and J. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems. *Structural optimization*, 14(1):63–69, 1997.

[8] I. Das and J. E. Dennis. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM J. on Optimization*, 8(3):631–657, Mar. 1998.

[9] K. A. De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD thesis, Ann Arbor, MI, USA, 1975. AAI7609381.

[10] K. Deb and H. Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *Evolutionary Computation, IEEE Transactions on*, 18(4):577–601, Aug 2014.

[11] K. Deb, M. Mohan, and S. Mishra. Evaluating the epsilon-domination based multi-objective evolutionary algorithm for a quick computation of pareto-optimal solutions. *Evol. Comput.*, 13(4):501–525, Dec. 2005.

[12] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Trans. Evol. Comp*, 6(2):182–197, Apr. 2002.

[13] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[14] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, pages 41–49, Hillsdale, NJ, USA, 1987. L. Erlbaum Associates Inc.

[15] J. H. Holland. Genetic algorithms. *Scientific American*, July 1992.

[16] J. Horn, N. Nafpliotis, and D. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, pages 82–87 vol.1, Jun 1994.

[17] K. Keeton, C. Santos, D. Beyer, J. Chase, and J. Wilkes. Designing for disasters. In *Proceedings of the 3rd USENIX Conf. on File and Storage Technologies*, pages 59–62, Berkeley, CA, USA, 2004.

[18] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, Jan. 2003.

[19] J. D. Knowles and D. W. Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evol. Comput.*, 8(2):149–172, June 2000.

[20] K. McClymont. Recent advances in problem understanding: Changes in the landscape a year on. In *Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '13 Companion, pages 1071–1078, New York, NY, USA, 2013. ACM.

[21] P. K. McKinley, S. M. Sadjadi, E. P. Kasten, and B. H. C. Cheng. Composing adaptive software. *Computer*, 37(7):56–64, July 2004.

[22] A. T. Peterson, J. Soberon, R. G. Pearson, R. P. Anderson, E. Martinez-Meyer, M. Nakamura, and M. B. Araujo. *Ecological Niches and Geographic Distributions (MPB-49)*. Monographs in Population Biology. Princeton University Press, 2011.

[23] A. J. Ramirez, D. B. Knoester, B. H. C. Cheng, and P. K. McKinley. Applying genetic algorithms to decision making in autonomic computing systems. In *Proceedings of the 6th International Conference on Autonomic Computing*, ICAC '09, pages 97–106, New York, NY, USA, 2009. ACM.

[24] V. Seydi Ghomsheh, M. Khanehsar, and M. Teshnehlab. Improving the non-dominate sorting genetic algorithm for multi-objective optimization. In *Computational Intelligence and Security Workshops, 2007. CISW 2007. International Conference on*, pages 89–92, Dec 2007.

[25] O. Shir. Niching in evolutionary algorithms. In G. Rozenberg, T. Back, and J. Kok, editors, *Handbook of Natural Computing*, pages 1035–1069. Springer Berlin Heidelberg, 2012.

[26] E. Zitzler and S. Kunzli. Indicator-based selection in multiobjective search. In *in Proc. 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII*, pages 832–842. Springer, 2004.

[27] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *Trans. Evol. Comp*, 7(2):117–132, Apr. 2003.