# MULTIOBJECTIVE OPTIMIZATION FOR SPACE SYSTEMS ARCHITECTURE: APPLYING AND EXTRACTING KNOWLEDGE

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Nozomi Hitomi

May 2018

MULTIOBJECTIVE OPTIMIZATION FOR SPACE SYSTEMS ARCHITECTURE: APPLYING AND EXTRACTING KNOWLEDGE

Nozomi Hitomi, Ph.D.

Cornell University 2018

Distributed spacecraft missions (DSM) are gaining traction in the space community for the potential to deploy multiple simple and low-cost spacecraft to provide high temporal resolution of observations over regions of interest. Designing a DSM, however, remains challenging due to the many architectural decisions involved at both the spacecraft-level and system-level and due to the presence of multiple conflicting objectives of maximizing performance while minimizing cost and risk. One proven approach to architecting DSMs is to gather a team of design experts from multiple disciplines and leverage their combined experiences to quickly identify a few feasible mission architectures. The rapid design process allows the team to refine the design problem several times to better capture the stakeholders' needs. A typical design team, however, only explores a handful of alternative missions, which limits their understanding of the key design decisions that determine a DSM's metrics. Another popular approach is to convert the design problem into an optimization problem and rely on search algorithms to explore more of the tradespace. In particular, multiobjective evolutionary algorithms (MOEA) have shown promise on DSM design problems, but they are considered computationally inefficient because they generally don't leverage the available domain- or problem-specific knowledge. To identify promising missions, MOEAs require evaluating hundreds or thousands of candidate solutions using computationally expensive simulations to compute their metrics. These computational burdens hinder designers from iterating through multiple problem formulations.

This thesis proposes a new tradespace exploration tool that combines the efficiency of expert design heuristics with the explorative power of an MOEA. The tool exploits the available expert knowledge to push the exploration to the most promising regions of the tradespace while searching other regions of the tradespace for novel solutions not captured by the knowledge. First, expert design knowledge is encoded as knowledge-dependent operators so that they can easily be incorporated into MOEAs. Next, an MOEA is augmented with an adaptive operator selection strategy (AOS) that allows an MOEA to efficiently utilize multiple evolutionary operators by constantly monitoring each operator's ability to create high-quality solutions and adapting the search strategy to apply the most effective ones. Given several knowledge-dependent operators along with conventional, knowledge-independent operators, the AOS can explore the tradespace by leveraging both the expert design heuristics and the explorative power of an MOEA.

This thesis also develops an MOEA that can extract new knowledge by applying a data mining algorithm to candidate solutions generated during an optimization run. This tool is useful when there are few or no design heuristics available for a given problem. The proposed tool encodes the extracted knowledge as evolutionary operators and uses them with an AOS to guide the remainder of the optimization process. The extracted knowledge is also provided to the user in an easy-to-understand form, with the hope that the information can help decipher the results and elucidate the key design decisions.

The efficacy of the proposed tradespace exploration tools are demonstrated on a DSM design problem for climate monitoring. The results shows that combining an MOEA with knowledge from experts or data mining algorithms leads to significant improvements in computational efficiency over a conventional MOEA.

## BIOGRAPHICAL SKETCH

The beginnings of Nozomi Hitomi's path toward engineering can be traced back to when he was 4 years old in Cleveland, Ohio. One day at daycare, the class went to "Safety Town" and learned from the local police officers that you should always wear your helmet when riding a bicycle because it can protect your head against hard impacts. When Nozomi got home later that day, he found his helmet that his parents made him wear when riding. As if he was challenging the foundations of the advise given earlier that day, Nozomi strapped on his helmet, ran from one end of the hallway with his head down, and slammed his head into the drywall at the other end. Although there was a 8 inch hole in the wall that Nozomi would be scolded for, he learned that helmets indeed protect your head.

In successive years, Nozomi grew to enjoy making things. He loved playing and building with Legos and K'NEX, and a electric train set, where the tracks could be laid in any which way. Sometimes, designs were a bit unconventional like taping tracks for Hot Wheels cars together to create vertical loops or carving a semi-truck for the Pine Wood Derby event in Cub Scouts when everyone else carved a sleek-looking car. For his science fair project in 7th grade, he tested the effects of centrifugal forces on plant growth using styrofoam soup bowls spun by a magnetic stir plate. Quirky ideas followed him to Dartmouth College, where his freshman design team worked on a recycling bin that could sort materials based on the sound of an object thrown into the chute of the bin.

Upon graduating Dartmouth with a B.A. and B.E. in mechanical engineering in 2011, Nozomi worked as a design engineer at Honda R&D Americas in Columbus, OH. At Honda, he was responsible for designing the roof structure of several models taking into account new safety standards from NHSTA and manufacturing processes being installed at the factory. The major projects that he was involved in were the 2014 Acura TLX, 2016 Acura NSX, and 2016 Honda Civic. These models required designing from the ground up or major re-designs, and although Nozomi was able to use his creativity to contribute innovate solutions, he did not always have an easy time implementing certain designs. The Civic posed the greatest challenges, where Nozomi involved in the early design stages. He was responsible for guaranteeing that the stylists reserved enough packaging space in the areas critical for safety. It was a constant battle with the stylist who wanted to protect their aesthetic visions. Other challenges on the job included significant design changes due to decisions made late in the design process to change available features. Such incidences typically resulted in headaches for many, but inspired Nozomi to explore other design methodologies.

In the Fall of 2013, Nozomi moved to Ithaca, NY to begin his Ph.D. degree in mechanical engineering at the Sibley School of Mechanical and Aerospace Engineering at Cornell University. Coincidentally, Daniel Selva was looking for his very first Ph.D. student to research tools to support the design process, especially in the early phases of design for space missions. Seizing this perfect opportunity to address his frustrations from the design process at Honda and to get closer to fulfilling a childhood dream to work in the space sector, Nozomi began work in the Systems Engineering Architecture and Knowledge (SEAK) lab under Dani Selva.

Outside of his design career and research, Nozomi enjoys being outside. He likes to go on short day hikes, cycle, and most of all, go fishing. He took a fly fishing course through Cornell's physical education program, and although he only caught a few bluegill during the course, Nozomi eventually learned when and where to fish during his time in Ithaca. He has caught many of the game species in the area, the largest being a 33 inch lake trout weighing 8 pounds. In addition to outdoor hobbies, he likes to cook, brew, and eat with friends.

ACKNOWLEDGEMENTS

I am grateful to the many who have journeyed with me during my time in Ithaca and supported me in my Ph.D. endeavors. None of the work in this thesis would have been possible without the guidance of Dani Selva. I feel very lucky to have started at Cornell just as Dani was looking for his first batch of Ph.D. students and that our research interests and personalities were compatible. Through many one-on-one sessions, he taught me everything I now know about systems architecture and helped troubleshoot (even debug) my algorithms and methodologies. I am thankful for the many hours that Dani put into revising and critiquing my lengthy conference and journal papers, and I am a better writer and researcher because of it. I also cherished the lab Dani put together and its culture that made the Selva lab feel like a family. I need to thank his wife Ana Diaz and his daughter Emma for sharing Dani and his time.

My other thesis committee members also played a great role in shaping me and my research. My first ever aerospace engineering course was Mason Peck's spacecraft dynamics class, and learning about attitude determination and controls furthered my understanding and appreciation of spacecraft design. Pat Reed taught me much of what I know about multiobjective optimization and evolutionary algorithms, and he really helped me refine my ideas on adaptive operator selection.

I am also thankful for my committee members connecting me to practitioners and researchers outside of Cornell. My work on the TROPICS mission with Bill Blackwell at Lincoln Laboratory provided me with an exciting opportunity to apply tradespace exploration for a real-problem. Working on TROPICS led to further opportunities with Jacqueline Le Moigne and her TAT-C team who took interest in my research and awarded me with a NASA Science and Technology Research

Fellowship (NSTRF) that funded my research for the last two years. Through this fellowship and work on TAT-C, I was encouraged that my research could contribute to NASA's design tools. I also was able to learn about and talk to people working at the Goddard Mission Design Lab and gain a better understanding for the breadth of users and use cases for a mission design-support tool. My internship at the Aerospace Corporation was also very enlightening and reassured me that interest in tradespace exploration tools was not isolated to just NASA. I want to thank Matt Ferringer, Lake Singh, Ron Clifton and Anne Gick for providing me with the opportunity to combine the methods and tools developed at Aerospace with parts of my research.

I need to thank all of the friends I made in Ithaca that made it possible to get through the Ph.D. program. Bryan Peele and Christoforos Mavrogiannis got me through the torrents of dynamics and controls homework that often required late nights and Friday beers. The others who kept me sane and smiling through the years are Ravi Patel, Ilse Van Meerbeek, Jon Jalving, Yolanda Lin, Robert Carson, Jack Hartman, Emily Powell, Bo Yang, Jordan Chipka, Zimu Zhu, Suwon Bae, Joyce Fang, George Valogiannis, and Matt Kasemer. Some of my best times at Cornell were sharing food and drink and laughing with all of you.

Thank you to my lab mates who made time in lab infinitely better than just working at my desk Nathan Knerr, Harris Bang, Clara Abello, Pau Buzzi, Ximo Gallud, Samalis Santini, Antoni Viros, Prachi Dutta, and Filipe Pereira. I particularly enjoyed my cultural and religious education I received from you.

I couldn't have completed my thesis work without the support and love from my family. I never really appreciated my parents' research when I was young, so I am extra grateful for my father's interest in my research and my mother's encouragement to pursue my dreams. My favorite new family tradition that started

in first full year in Ithaca is our family video-chats for birthdays. All of us would sing "Happy Birthday", but due to lags in our connections, it always turns out as a hysterical, off-key and offbeat song. Even though we're spread out across America now, silly moments like these warm me up and make you feel closer. Thank you Yutaka, Emi, and Kei for being my goofy and loving siblings.

Lastly, I am immensely grateful for the love and support from Jen Fownes over the last 10 years. From our time together at Dartmouth, she has been a source of much of my happiness and encouragement. She put up with my hectic work schedule during my first year at Cornell, when I would come home to a warm dinner before heading back to campus to finish work. She introduced me to salsa, which is now in my repertoire of other ballroom dances that she has exposed me to. I'm not a great dancer, but it is a fun way to show off a little at weddings and other formal events. I am lucky to have found some one who laughs at my bad jokes, enjoys eating as much as I do, and dreams of new adventures together.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

CHAPTER 1

# INTRODUCTION

## 1.1 Motivation

In recent years, interest in distributed spacecraft missions (DSM) have been gaining traction [2, 159] because of their advantages over a monolithic missions and the recent advances in space technology that are making DSMs more economically feasible. Compared to a monolithic mission, a DSM can fly multiple simpler and inexpensive spacecraft [47, 143] and provide increased capabilities such as improved temporal, spatial, and angular sampling [29, 135, 136]. Funding a DSM has historically been cost-prohibitive, but the recent popularization of standards such as CubeSat specifications [149], development of smaller and cheaper instruments and bus components [166], and availability of more economical launch opportunities [159] are creating new possibilities for DSMs.

As a result government agencies and the private sector are now seriously exploring and investing in DSM missions. One example is NASA's CYclone Global Navigation Satellite System (CYGNSS) mission, which launched in 2017 and is tasked with monitoring wind speeds of tropical cyclones with 8 spacecraft [154]. Previous monolithic missions, such as ASCAT and QuikSCAT, measuring ocean-surface wind speeds have a mean revisit time of over 24 hours in the tropical regions [160]. In comparison, CYGNSS can sample a tropical cyclone with a mean revisit time of 4 hours [156], drastically increasing the ability to not only predict a storm's evolution, but also to accurately model storm genesis and intensification. In 2016, NASA funded another DSM called Time-Resolved Observations of Precipitation structure and storm Intensity with a Constellation of Smallsats

(TROPICS) to provide all-weather observations of 3D temperature and humidity profiles of tropical storms with 12 CubeSats [12]. TROPICS is expected to achieve median revisit times of 60 minutes or less in the tropics, which will advance our understanding of tropical storms and develop more accurate numerical weather prediction models. In the private sector, Planet Labs has been on the forefront of investing in DSMs having launched 149 satellites as of 2017 [1]. Their goal is to provide medium-to-high resolution imaging of the entire planet on a daily basis and deliver Earth imagery and imagery-derived data products to customers involved in agriculture, defense, emergency response, and insurance among other markets [1, 14].

DSMs have a lot of potential in many domains, but due to their complexities in design and operation, any DSM's success will depend on its system architecture. The architecture of a system is "an abstract description of the entities of a system and the relationship between those entities" [31], and it represents the high-level decisions made by system engineers and architects. All complex systems involve many stakeholders and different interacting components, and it is important to establish an appropriate architecture in the early phases of the design process because architectural decisions have large impacts on downstream decisions and on the system's performance, cost, risk, flexibility, and other metrics [31, 42, 155, 161]. Examples of architectural decision for a DSM include the number of spacecraft to utilize for the mission, their individual instrument payloads, and their assigned orbits.

A key step in selecting an appropriate system architecture is tradespace exploration, where many candidate concepts are generated and evaluated to understand the key tradeoffs and drivers of the metrics. Tradespace exploration for DSMs, however, is challenging due to: 1) the large number of design variables including each satellite's orbital parameters, payload, bus, and launch vehicle and their complex, nonlinear interactions; 2) many nonlinear constraints involving packaging space, power requirements, link budgets, thermal control, and on-board data handling capabilities; 3) multiple conflicting objectives of maximizing performance while minimizing cost and risk; 4) and the computationally expensive models used to simulate and evaluate a given mission.

Currently, there are two main approaches for handling the complexity tradespace exploration for DSMs. The first approach, adopted by concurrent design teams such as Team X at NASA JPL, is to rely heavily on a team of design experts to quickly generate and evaluate candidate missions and to rapidly iterate through multiple design formulations [42, 87, 142, 198]. Leveraging their combined expertise, design heuristics, model-based simulation tools, and effective communication within the team, a concurrent design team can rapidly identify a few of the most promising missions within a vast tradespace. The quick turn-around allows the team to iterate and explore new design formulations that take into consideration new information gained from previous formulations. By iterating through multiple formulations, the team is not only more knowledgeable about the particular mission at hand, but also more confident that the mission selected at the end of the process will meet the stakeholder needs [87, 119]. This human-centric approach, however, requires a lot of human resources, years of design experience in multiple disciplines, and often a dedicated facility or center to promote an interactive and collaborative design process [87, 142]. Moreover, the experts' design heuristics that quickly reveal promising mission architectures can limit and bias the concept generation to only the seemingly promising regions of the tradespace and ignore any novel solutions that may exist in other regions [155]. Finally, with limited time and effort, these concurrent design teams can analyze only a dozen or so concepts in-depth before selecting a final mission architecture to develop in detail [29, 90].

The other approach is to formulate the design problem as an optimization problem [31, 90] and use a computer algorithm to explore the vast tradespace of possible DSMs and discover promising missions. In particular, multiobjective optimization algorithms such as multiobjective evolutionary algorithms (MOEA) are playing a critical role as a decision support tool in architecting and designing DSMs [24, 50–53, 75, 79, 127, 163, 165, 177, 177, 184]. Unlike single-objective optimization methods, multiobjective optimization approaches explicitly allow tradeoff analysis because they keep each objective separate and present the architect with promising solutions that optimally trade the multiple system metrics such as performance, cost, and risk. Moreover, MOEAs can evaluate thousands or millions of alternative missions and provide insights of the key trades and drivers through multiobjective visual analytics [25, 26, 100, 104, 189, 191, 192] and data mining algorithms [7, 37, 40, 141]. Compared to a human-centric tradespace exploration, conventional MOEAs utilize a stochastic sampling process that is not guided by expert knowledge, resulting in exploration of seemingly unpromising areas of the tradespace and an opportunity to discover novel solutions. The disregard for problem- or domain-specific knowledge, however, results in an computationally inefficient search that relies on evaluating many solutions before discovering a set high-quality solutions. Evaluating many thousands or millions of alternative missions is impractical when architecting a DSM because evaluating even a single mission requires computationally expensive model-based simulations that can take minutes, hours, or days to complete. Consequently, tradespace exploration with MOEAs on one problem formulation is time intensive, and iterating through multiple formulations becomes challenging. The inability to iterate through multiple problem formulations greatly inhibits the problem discovery process [188,190] that helps to refine model assumptions and arrive at an appropriate problem formulation.

Table 1.1 summarizes the advantages and disadvantages of these two paradigms for architecting DSMs. Note that the table is not comprehensive, and thus, the number of advantages versus the number of disadvantages for a particular method does not imply that a certain method is better or worse. Instead, Table 1.1 is meant to show that the two methods have many complimentary features. On the one hand, expert-based tradespace exploration is able to quickly identify promising missions and iterate through multiple problem formulations, but it is resource and knowledge intensive and can only explore a small number of alternative mission concepts. On the other hand, an MOEA can explore a vast area of the tradespace using little design expertise, but its computational inefficiency makes it difficult for mission architects to iterate through multiple problem formulations.

This thesis aims to develop a tradespace exploration tool for architecting DSMs that can explore a vast region of the tradespace and identify promising mission concepts quickly. To achieve this goal, the tool brings together the reasoning capabilities of humans with computationally powerful algorithms. Specifically, the efficient and powerful design heuristics used in expert-based tradespace exploration are combined with the explorative capabilities of MOEAs to help system architects design DSMs. This thesis demonstrates the decision-support tool's capability to not only efficiently explore the tradespace and identify high-quality solutions, but also to reveal new insights into the design problem that can inform decision making and future tradespace exploration.

Table 1.1: Advantages and disadvantages of expert- and MOEA-based tradespace exploration

| | Advantages | Disadvantages |
|---|---|---|
| Expert-based | • Quickly identifies promising missions<br>• Enables multiple problem formulations | • Requires years of experience<br>• Limited tradespace exploration<br>• Neglects seemingly unpromising solutions |
| MOEA | • Requires little design expertise<br>• Able to explore millions of missions<br>• Explores seemingly unpromising solutions | • Computationally inefficient<br>• Does not leverage expert knowledge<br>• Hinders problem discovery |

The remainder of this Introduction chapter is organized as follows. Section 1.2 presents the necessary background on tradespace exploration, multiobjective optimization problems, MOEAs, prior work on incorporating knowledge into MOEAs, limitations of existing approaches, and the research gaps. A general problem statement is given in Section 1.3, and Section 1.4 presents the approach for developing an efficient and effective decision-support tool. Section 1.5 poses the research questions that are addressed in this thesis. Lastly, Section 1.6 lays out the structure of the rest of the thesis.

## 1.2 Background

### 1.2.1 Tradespace Exploration

The set of possible architectures in a space defined by two or more metrics is called the tradespace [31, 155]. For real-world design problems, it is often challenging to identify an architecture that appropriately balances multiple conflicting metrics, and designers strive to find a good tradeoff in all metrics. A common trade seen in many systems including DSMs is between some performance metric (e.g., revisit time) and cost, where an increase in performance will generally increase the cost of a system. The relationship in this trade is typically not uniform throughout the tradespace; it might be possible for architectures in one region of the tradespace to significantly improve their performance with little additional cost while architectures in another region of the tradespace require substantial cost increases for a modest or no improvement in performance.

An architect is responsible for understanding the characteristics and causes of such trends within a tradespace in order to move forward with a final architecture or set of architectures for further analysis. An effective approach to understanding the tradespace is through the Design by Shopping approach [6]. In Design by Shopping, designers or architects are presented with several to many candidate architectures to build a better intuition of the tradespace, and based on the presented architectures the architect can "shop" for or choose an architectures to analyze further. An architect can make an informed decision by relying on visualization tools [25, 26, 100, 104, 115, 141, 180, 188, 192] or data mining algorithms [45, 75, 88, 95, 164] to analyze the candidate architectures and gain valuable information about the attainable limits of each metric, the tradeoffs between metrics, the key design decisions that impact the tradeoffs, and how one architecture compares with others. In the early phases of the design process, Design by Shopping is preferred over point solution comparisons, which compares two or three specific design solutions in detail [31]. Point solution comparisons provides an architect with incomplete knowledge about the tradespace and can prematurely focus the design process on a specific solution without considering other, potentially better, alternatives [155].

In order to conduct Design by Shopping, a set of candidate architectures must be generated and evaluated to compute their metrics. A sufficient number of candidate architectures needs to be generated in different regions of the tradespace such that the architect can draw accurate and meaningful conclusions about which architectures are preferable. This solution generation process for DSMs is challenging because 1) the size of the tradespace is large [90, 158, 163] and 2) evaluating a single architecture can be time and computationally expensive [51, 53]. The size of the tradespace and expensive evaluations preclude solution generation methods such as full or fractional factorial enumeration that sample the entire tradespace and generate too many solutions [167]. Instead, tradespace exploration methods are used to generate and evaluate solutions in the most promising regions of the tradespace and avoid generating low-quality solutions that have poor tradeoffs in the metrics. To achieve this, recent tradespace exploration tools focus on converting the design problem into a multiobjective optimization problem [31] and utilizing a multiobjective optimization algorithm [51, 53, 54, 75, 79, 90, 118, 153, 165, 183, 191].

### 1.2.2 Multiobjective Optimization Problems

Multiobjective optimization problems (MOP) can capture many real-world design problems that involve optimizing multiple conflicting and incommensurable objectives [28]. Formulating a problem with multiple objectives is particularly useful when the decision makers' preferences in the objectives are difficult to define with a utility function (e.g., a weighted sum of multiple objectives) that collapses the multiple objectives into a single objective. Under the Design by Shopping paradigm, the goal of multiobjective optimization is to simultaneously optimize in all objectives, present the decision makers with a diverse set of solutions that optimally trade the objectives, and allow the decision makers to develop their preferences based on the presented solutions and their tradeoffs.

Without loss of generality, it is assumed that all objectives are to be minimized and describe a continuous or discrete multiobjective problem as:

$$\min_{x \in X} f(x) \text{ subject to}$$
$$g_i(x) \leq 0, \;\; i = 1, 2, ..., p$$
$$h_j(x) = 0, \;\; j = 1, 2, ..., q$$

where $x$ is a design vector containing $L$ real- or discrete-valued design variables, $X \subseteq \mathbb{R}^L$ is the decision space that can include continuous or discrete design variables, $f(x) : X \rightarrow \mathbb{R}^m$ is the objective function mapping $x$ into the $m$-dimensional objective space, $g_i(x)$ is the $i$th inequality constraint, $p$ is the number of inequality constraints, $h_j(x)$ is the $j$th equality constraint, and $q$ is the number of equality constraints.

For an MOP, two solutions can be objectively compared using Pareto dominance. Given two solutions with objective vectors $u$ and $v$, $u$ is said to dominate $v$

if and only if $\forall i \in \{1, 2, ..., m\}$ $u_i \leq v_i$ and $\exists j \in \{1, 2, ..., m\}$ such that $u_j < v_j$. "$u$ dominates $v$" is expressed as $u \prec v$, where $\prec$ is an extension of $<$ in $\mathbb{R}$ [200]. Given a set of solutions $S$, the subset $s \subseteq S$ that are not dominated by any other solution in $S$ are called the non-dominated solutions, also referred to as the Pareto front $PF$ of $S$. The truly optimal set of solutions is called the true Pareto Front $PF^*$, which consists of feasible solutions that are not dominated by any other solution belonging to the set of all possible solutions $X$. In practice, however, exactly determining $PF^*$ is difficult or impossible, so the ultimate goal when solving a multiobjective problem is to obtain a set of solutions whose Pareto front $PF$ approximates $PF^*$. Since $PF^*$ typically has a diverse set of solutions, a multiobjective optimization algorithm not only has to push convergence of $PF$ toward $PF^*$, but also must obtain diverse solutions in $PF$ that spread across $PF^*$.

### 1.2.3 Multiobjective Evolutionary Algorithms

Real-world problems often have multiple incommensurable objectives and have unfavorable mathematical properties (e.g. non-linear, discontinuous, non-differentiable, non-convex) in the objective and constraint functions $f, g,$ and $h$. These properties make it difficult or impossible to apply efficient optimization algorithms, such as gradient descent algorithms, without making strong simplifying assumptions about the problem. Instead, many practitioners tackle real-world problems with heuristic and meta-heuristic optimization algorithms including multiobjective evolutionary algorithms (MOEA) [22, 24, 50–53, 75, 79, 106, 116, 118, 127, 153, 163, 165, 177, 177, 184].

MOEAs are a popular choice to solve real-world problems because they can not only handle objective functions that are multimodal, nonlinear, and nonconvex, but also optimize all objectives simultaneously. MOEAs are inspired by natural evolution, where "fit" individuals are able to pass on their genetic information to their offspring. In an MOEA, a solution to the problem is represented by an individual, and a set of individuals create a population. The individuals undergo asexual and sexual reproduction via one or more mutation and crossover operators to create new offspring that enter the population. To maintain a specified population size and improve the overall quality of the population, a selection operator tends to select the fittest individuals to remain in the population. Occasionally, the selection operator will allow less fit individuals to remain in the population to help maintain diversity within the population and avoid excessive selection pressure, which can cause the search to prematurely converge on local optima. Over time, an effective MOEA will evolve a population of solutions toward more fit individuals that contain good trades in the objectives.

MOEAs are versatile and can be applied to problems across many domains, and they have shown much promise on space-related problems including optimizing satellite constellation configurations [50–53,177,184] and the architecture for Earth observing satellite systems [163,165]. Conventional MOEAs such as NSGA-II [39] or MOEA/D [193], however, are considered computationally inefficient because they stochastically sample the search space and require many function evaluations to identify a high-quality solution set [22, 118]. If each function evaluation is computationally expensive and takes hours or days, then evaluating thousands of solutions becomes intractable.

An MOEA's inefficiency can in part be explained by their failure to leverage any available knowledge about the problem structure or the domain [13,22,118] or take advantage of good design features that can be explicitly extracted by data mining the solutions discovered during the optimization. It is well known from the No Free Lunch Theorem (NFLT) for optimization that general purpose algorithms like MOEAs cannot perform well across all possible problems [186,187]. The NFLT for optimization, in short, says that if no prior assumptions are made about the problem at hand, no optimizer can be expected to perform better than any other [80]. In other words, to improve an algorithm's computational efficiency and its ability to consistently discover high-quality solutions, it must be specialized to a specific problem or class of problems by incorporating problem- or domain-specific information [13]. Note that this does not violate the NFLT for optimization because the increased efficiency and performance of the specialized algorithm is offset by its decreased performance on problems outside of its specialization. The decreased efficiency and performance on the other problems, however, is of little importance as long as the algorithm is applied to problems that are relevant to the provided information. The following subsection discuss the existing literature on knowledge-intensive EAs, which incorporate problem- or domain-specific knowledge into EAs to significantly improve their search performance and efficiency.

### 1.2.4 Knowledge-intensive EAs

Even prior to the formal proofs of the NFLT for optimization, the pioneers of evolutionary computing advocated for the incorporation of domain-specific knowledge into EAs and other "black-box" optimization algorithms [33, 66, 129, 152]. Early work focused on genetic algorithms and Schema Theory [81] and recommended using domain-specific knowledge to obtain good solution representations, where the schema or "building blocks" [66] were meaningful to the underlying problem and could be passed onto offspring solutions. Specialized solution representations can also help by reducing the size of the search space if they can remove redundant representations of a given solution [49]. Furthermore, if the problem involves constraints, it is beneficial to use a representation that reduces or eliminates the generation of infeasible solutions [13, 27, 49, 128]. However, defining a specialized representation for the problem at hand can be difficult [74], so it is not always a feasible method of incorporating knowledge.

Expert knowledge can also be leveraged to create a low-fidelity evaluation model to approximate the fitness of a solution [91] and inform initialization, selection, mutation, or crossover [92]. For example, a 3D finite element analysis can sometimes be approximated using a less costly 2D analysis. If a good approximation model does not exist for the problem of interest, however, it may be time consuming or intractable to create a low-fidelity model from first principles or from available training-data. Moreover, it is difficult to construct an approximate model that is globally correct and does not introduce false optima that can mislead the search [91, 92].

A simpler and popular alternative is to intelligently initialize the population with known high-quality solutions as opposed to random initialization or statistical methods such as Latin hypercube sampling [98]. Seeding a population with a few known good solutions can "warm start" the algorithm and significantly reduce the time required to discover high-quality solutions [63, 118, 172]. Initializing the population with known feasible solutions can also be beneficial for solving constrained optimization problems because it reduces the time spent on finding a feasible region of the design space [13, 168]. Intelligent initialization must be implemented with caution, however, so that it does not significantly reduce the diversity of the population and hinder the search [168, 172]. Moreover, intelligent population initialization only applies knowledge at the very beginning of the search, and the algorithm does not explicitly use that knowledge to conduct the rest of the search.

In contrast, KD operators or KD constraints can be used to continuously guide the search process. In addition, both approaches can accommodate multiple sources of knowledge, which is desirable when solving complex design problems that benefit from leveraging a lot of available knowledge from expert designers or multiple rules learned by a KDO algorithm. Despite their advantages, however, there are three main issues that must be addressed to successfully apply KD operators and KD constraints. First, in the presence of multiple sources of knowledge, an EA should be able to detect and focus on applying the knowledge that is most beneficial in improving the search performance. Not all knowledge will be equally beneficial in accelerating the optimization [77], but prior to the search, it can be difficult to assess how well a KD operator or constraint will help guide the search. Moreover, there is some evidence in the literature showing that incorporating knowledge can actually fail to improve an EA's search performance [22]. Second, multiple sources of knowledge can introduce conflicting information, especially when trying to solve multiobjective problems because many design heuristics focus on improving a single objective. For example, in a spacecraft design problem maximizing scientific benefit and minimizing cost, one heuristic suggests grouping several instruments together on the same spacecraft to leverage data fusion opportunities and increase the scientific benefit, but another heuristic advocates for fewer onboard instruments to reduce the spacecraft weight and cost. These heuristics are in direct conflict with each other, but different solutions from separate regions of the tradespace can be improved using one or the other. Therefore, an EA should have a mechanism to balance the use of these heuristics to achieve good tradeoffs in the conflicting objectives. Finally, an EA should not over-rely on the provided knowledge because it can introduce a strong bias that leads the EA to prematurely converge on local optima [169, 172]. Therefore, an EA should not only exploit the knowledge to guide the search toward the promising regions of the tradespace, but also explore the tradespace to potentially discover novel solutions that are not captured by the provided knowledge.

The following two subsections provide the details of related work on EA's that utilize KD operators and KD constraints and their mechanisms to address the three issues above. It should be noted that while we use the term KD constraints in this thesis, a solution that does not satisfy a KD constraint is not necessarily an infeasible solution to the problem at hand. For example, a KD constraint may penalize a solution for a satellite if its payload contains too many instruments, which can result in low duty cycles due to limited onboard power and deliver low scientific benefit. Such a satellite is still a feasible solution, however, and may even be preferred for the data fusion opportunity on the measurements taken at the same time and from the same perspective. Such KD constraints are meant to guide the search to promising solutions but not change the underlying problem formulation. So, to remove possible confusion with regard to the feasibility of a solution, a solution that satisfies a KD constraint will be referred to as *consistent* as opposed to feasible. Similarly, a solution that does not satisfy a KD constraint will be referred to as *inconsistent* instead of infeasible. Lastly, analogous to constraint violations, the amount by which a solution violates a KD constraint will be quantified and called a *knowledge violation*.

## Knowledge-dependent operators

Early work on KD operators focused on exploiting the problem structure to preserve the feasibility of solutions, specifically for combinatorial problems dealing with scheduling or permutations [49, 152]. If the problem structure cannot be fully exploited to develop operators that maintain feasibility, repair operators can be employed to remedy a solution's infeasibility [128, 130]. These operators that preserve the feasibility of the solutions eliminate or reduce the need to search for feasible solutions and allow the algorithm to spend the entire search within the feasible region.

More commonly, knowledge-intensive EAs leverage domain-specific heuristics or knowledge of features common in high-quality solutions. For example, Bonissone et al. create specialized mutation operators to improve lamp filters. They focus on modifying the design variables known to significantly impact the efficiency and color temperature of the transmitted light and modify the design variables to create smooth transitions in transmittance over the visible spectrum [13]. Similarly, Chabuk et al. exploit a known causal relationship where an impedance mismatch between the transmission lines and radiating elements of a dipole antenna is caused by a suboptimal antenna length [22]. Solutions with this impedance mismatch are subject to larger mutations on the antenna length and are less likely to pass on the gene associated with antenna length when recombined with another solution. Mahbub et al. use knowledge of the gross effect of a design variable for a mixed-energy portfolio on the objectives (e.g. increasing the pro portion of natural gas generally increases emissions and reduces cost). The knowledge is used to bias the mutation to prefer increasing or decreasing design values to improve specific objective values [118]. Calvo et al. search for protein structures with minimal free energy by using mutations that alter the angles formed in their secondary and tertiary structure such that they are consistent with known angle dependencies in similar proteins [18]. Finally, Hitomi and Selva utilized several KD operators on a multi-satellite design problem that, for example, added specific instruments to a satellite to leverage cross-registration of data products or removed superfluous instruments from its payload [77].

While the above have demonstrated that KD operators can help improve an algorithm's search performance, over-utilizing them can cause a reduction in the population's diversity and lead the search to prematurely converge on local optima [169]. KD operators drive the search to regions of the tradespace that are known to contain promising solutions, but this bias can be too strong and undermine the exploration of the tradespace. One solution is to apply both knowledge-independent and KD operators together, and increase and decrease their rates, respectively, according to a predetermined schedule [18, 88, 118]. This allows the algorithm to exploit the given knowledge at the beginning to push the search toward promising regions of the tradespace, and then focus on exploration at the end of the search to avoid premature convergence. However, determining an appropriate schedule prior to the search is not straightforward because the performance of many good operators, including knowledge-independent ones, depends on both the particular instance of a problem and on the state of the search [74, 78]. Moreover, determining an appropriate schedule becomes more complicated in the presence of operators encoding conflicting knowledge because they can undo each other's modifications.

A more adaptive procedure is used in hyperheuristics [17,30] and adaptive operator selection (AOS) strategies [56, 78]. Given a set of operators, these approaches continuously adapt their search strategy by focusing on utilizing the best subset of operators that consistently discovers improving solutions. Hyperheuristics and AOS strategies are often used to increase the robustness of the search to the given problem or algorithm parameters [17,72,78,109], but they have further advantages when incorporating domain-specific knowledge [77]. These methods allow users to incorporate multiple KD operators instead of only a few (1-3 operators) in other approaches [13,18,22,118]. Even in the presence of conflicting knowledge, the algorithm will focus on applying the knowledge that leads to better solutions, given the current state of the search. Furthermore, knowledge-independent operators such as crossover and mutation can be employed alongside KD operators, so if or when the KD operators are unable or no longer able to create improving solutions, the algorithm can adapt its search strategy to a more conventional evolutionary search. Thus, the adaptive selection over the given operators balances the exploitation of the provided knowledge with the exploration of other regions in the tradespace.

### Knowledge-dependent constraints

Despite the plethora of literature on constraint-handling methods for EAs [27, 128, 157], the literature contains few examples of applying knowledge through constraints. The existing constraint-handling methods are purposed for solving constrained-optimization problems, where it is assumed that the constraints are part of the problem definition and must be satisfied. Therefore, existing constraint-handling methods would treat KD constraints as an absolute truth even though expert knowledge and heuristics may not actually be accurate or relevant for a given problem instance. Moreover, additional constraints effectively reduce the size of the search space and prevent the exploration of novel solutions in seemingly unpromising regions of the tradespace [155]. Finally, conflicting knowledge in the form of constraints can create contradictions that always lead to inconsistent solutions, and the constraint-handling methods will guide the search to solutions with fewer knowledge violations as opposed to solutions with better objective values. For example, one KD constraint might limit the mass of a satellite to avoid using large and expensive spacecraft buses and launch vehicles, while another KD constraint forces a satellite to carry a minimum number of instruments to facilitate data cross-registration and leverage economies of scale. If the minimum number of instruments is large enough, then it can cause the satellite to exceed the upper limit on its mass. In this case, obtaining a solution that is consistent with both KD constraints becomes impossible.

Nevertheless, constraints provide a natural and simple way of guiding the search process to known good regions of the tradespace using existing constraint-handling methods. The Learnable Evolution Models (LEM) algorithm combines evolutionary algorithms with data mining to extract knowledge during the optimization and apply it to guide the remainder of the search. The knowledge is encoded as a logical rule in disjunctive normal form (DNF), where each literal in the rule represents a feature commonly appearing in good solutions [132]. This rule is applied to create new solutions that satisfy at least one literal in the rule. Since a solution only needs to satisfy one literal, conflicting knowledge can be incorporated into the DNF rule. However, the LEM does not have an adaptive mechanism to select the most appropriate literal or set of literals that a solution should satisfy to improve upon the best solutions found so far. This is especially problematic when the search moves to a region of the tradespace where all solutions are consistent with one of the KD constraints. If the newly created offspring also lie in the same region, then they will also satisfy the DNF rule, and none of the other KD constraints will help guide the search until the search moves away from that region.

A more adaptive constraint-based approach is seen in a work by Gaur and Deb [65]. A relationship between design variables is enforced only if it is present in a user-specified fraction of the non-dominated solutions in the current population, assuming that other high-quality solutions will also contain the same relationship. Thus, a KD constraint is enforced or activated as the algorithm discovers improving solutions that are also consistent with the knowledge. However, this approach does not have a mechanism for handling conflicting knowledge. Non-dominated solutions in separate regions of the tradespace can be consistent with different conflicting KD constraints. As a result, it is possible with Gaur and Deb's method that conflicting KD constraints are simultaneously enforced, which will produce solutions that are inconsistent with at least one of the KD constraints.

### 1.2.5 Knowledge Driven Optimization

Many knowledge-intensive EAs assume the knowledge is readily-available prior to the search, but often, it must be acquired by soliciting expert designers or gathering known, high-quality solutions, both of which can be time-consuming and complicated processes. Optimization algorithms that hybridize machine learning with evolutionary algorithms can bypass this human-centric, knowledge-acquisition process by extracting useful information from a set of known solutions or solutions discovered during the optimization. Bayesian optimization and Estimation of Distribution Algorithms are examples that use solutions discovered during the optimization to obtain knowledge in the form of a probabilistic model, which predicts the objective values and their associated uncertainties as a function of the decision values [23,62,101,146,197]. The probabilistic models inform the algorithm on how to sample the decision space to create new solutions that are expected to improve the current objective values and/or reduce the uncertainty in the model. Although these probabilistic models store useful knowledge, the knowledge they store is implicit in their models and is generally too complex for a human to interpret and understand. Therefore, the user has difficulty gaining new insights into the problem that can be easily communicated to other designers or stakeholders and documented for use on similar problems in the future [7, 89, 126].

Knowledge-driven optimization (KDO) approaches focus on extracting easy-to-understand design features from high-quality solutions or non-dominated solutions under the assumption that solutions near or on the Pareto front share certain properties that make them optimal [40]. By analyzing these properties, a system architect can not only have more confidence in the evaluation model but also gain valuable insights into the problem relating the decision variables and objective values. For example, on a hybrid rocket design problem, Watanabe, Chiba, and Kanazaki found that 83% of all Pareto-optimal designs had similar port radii [180], so when designing hybrid rockets in the future, it would be beneficial to explore designs with a similar port radius. These important design features are often revealed off-line through data visualization methods applying clustering and manifold learning such as self-organizing maps [141], iso-maps [104], heat maps with cp-lines [25], Hyper-Radial Visualization [26], analysis of non-correspondence areas [192], and Cityplot [100]. However, the knowledge, implicit in the visualizations, is subject to human interpretation, and unambiguously communicating the knowledge to other designers or translating the knowledge to future problems can be challenging [89].

To this extent, other KDO algorithms focus on extracting knowledge in a more explicit form with a formal, mathematical representation that also resembles natural language (e.g. first-order logic) so that it can be clearly communicated and understood by humans [7, 89, 126]. Moreover, these KDO algorithms do not require a human in-the-loop, allowing for a more automated process, where the newly learned knowledge is codified and can be reapplied during an optimization to accelerate the convergence or stored for future use. Such KDO algorithms rely on data mining methods including classification rule mining [75], classification trees [45, 88, 95, 164], and Algorithm Quasi-optimal (AQ) learning [21, 131].

One shortcoming of current KDO algorithms is that they do not discriminate knowledge that is effective at accelerating the convergence of the search from knowledge that has little benefit to the optimization process. Instead, they combine the extracted knowledge into a single rule or apply the knowledge through constraints. Even among the knowledge that helps guide the search, it is likely that some lead to more improving solutions than others. In fact, the ability of the extracted knowledge to accelerate the search has been shown to vary during the optimization, often with diminishing efficacy as the search progresses [75]. As the knowledge is applied to solutions in the population, the entire population may begin to reflect the encoded design features, after which, the knowledge cannot contribute further modifications. Therefore, rigidly applying knowledge may waste computational resources and prevent the algorithm from sufficient tradespace exploration to discover new regions with high quality solutions, which can lead to convergence on local optima [77].

EMO/I is a KDO algorithm that applies linear regression to a log-linear model during the search to obtain a rule in the form of a power-law that captures common design features in non-dominated solutions. EMO/I then uses the power-law to "repair" solutions during the remainder of the optimization process [65], but there is no mechanism that prevents applying the power-law rule if it does not lead to improving solutions. A similar approach uses distance-based regression trees during the search to extract rules that differentiate solutions that are near and far from regions of interest, and the extracted rules are then imposed as constraints on any new solutions [140]. As constraints, each extracted rule is given equal importance in guiding the search, which prevents the algorithm from focusing on applying the rule, if any, that leads to more improving solutions. Learnable evolution models (LEM) [133] uses AQ learning to generate multiple logical rules that relate specific design features to the high-quality solutions in the population. The rules are combined into one logical sentence in a disjunctive normal form, and future solutions must satisfy the sentence by conforming to at least one of the rules. Since it does not matter which rule in the sentence is satisfied, LEM cannot differentiate the rules within the ruleset that are effective in accelerating the convergence from those that do not help guide the search. LEMMO [95] is a variation of LEM that uses C4.5, a decision tree induction algorithm, instead of AQ learning. Each positive rule produced by C4.5 is used to create a new solution, treating each rule as equally important in guiding the search.

## 1.3 General Problem Statement

If MOEAs are to tackle complex problems that have computationally expensive evaluation functions, it will be beneficial for them to leverage as much available knowledge. The existing knowledge-intensive EAs and KDO algorithms, however, do not fully address the three main concerns when incorporating domain- and problem-specific knowledge from multiple sources into MOEAs. These concerns are presented again below.

1. In the presence of multiple sources of knowledge, an EA should be able to detect and focus on applying the knowledge that is most beneficial in improving the search performance.
2. Using knowledge from multiple sources can introduce conflicting information since many design heuristics focus on improving only one of the multiple conflicting objectives. MOEA should have a mechanism to balance the use of these heuristics to achieve good tradeoffs in the objectives.
3. An MOEA should balance the exploitation of the provided knowledge with the exploration of regions of the tradespace that are not captured by the knowledge to push the convergence rate without sacrificing the discovery of novel solutions and converging on local optima.

Some of the existing knowledge-intensive MOEAs and KDO algorithms simply have no mechanism that allows multiple sources of knowledge to be incorporated, while others apply the provided knowledge too rigidly and generally do not monitor whether the design heuristics are helping to produce improving solutions. There is a need to develop a more flexible and adaptive algorithm that can incorporate knowledge from multiple sources, identify and apply the knowledge that leads to improving solutions even in the presence of conflicting information, and balance the exploitation of the provided knowledge with exploration of other regions of the tradespace.

## 1.4 Approach

This work proposes a flexible and adaptive mechanism for applying knowledge, available prior to the search or extracted during the search using data mining methods, to guide the optimization process and improve the efficiency and efficacy of an MOEA. A major component of this work relies on using knowledge-dependent (KD) operators and an adaptive operator selection (AOS) strategy. The KD operators are used to encode available knowledge into design heuristics that are used during the search to try to improve a given solution. For example, in satellite design, if an active instrument and passive instrument operating near the same spectral bands are assigned to the same instrument payload, then the instruments will experience noise in their measurements from electromagnetic interferences. This knowledge can be encoded into a KD operator that removes one of the two instruments from the satellite's payload to alleviate the situation. The AOS is used to control the application of multiple evolutionary operators through a *credit assignment* strategy, which rewards operators for creating improving solutions and an *operator selection* strategy, which selects and applyies the operators with many credits. Through an AOS, multiple KD operators can be assigned to an MOEA and used alongside multiple knowledge-independent (KI) operators such as conventional crossover and mutation operators. The efficacy of these operators on a given problem is dependent on the state of the search [74,78], so an adaptive selection of the given operators allows for the selection of the most effective operators at each iteration. Moreover, the AOS will help to balance the application of the provided knowledge with the exploration of other regions of the tradespace not captured by the knowledge. The performance of both KD and KI operators will be monitored, and the KD operators will be used more often only when the provided knowledge consistently leads to improving solutions. Otherwise, the AOS can adopt a more conventional evolutionary search with the KI operators that can better explore the tradespace.

For problems when knowledge is not available prior to the search process, a

KDO method can be utilized with the proposed approach using an AOS. New knowledge, extracted through data mining solutions discovered during the optimization, is encoded as new KD operators. These KD operators are utilized alongside KI operators by an AOS, and the search strategy is adapted to utilize the extracted knowledge if it beneficial to the optimization process.

## 1.5 Research questions

Prior to the work conducted in this thesis, no MOEA had been augmented with an AOS with the purpose of incorporating domain- and problem-specific knowledge. The literature on knowledge-intensive EAs focus on single-objective problems [106], discuss algorithms that do not have a mechanism to adapt the use of the provided knowledge [118, 162, 163], or both [13, 18, 22, 88]. The existing literature regarding AOS strategies explores the elevated generality of an EA and increased robustness of an EA's performance to its parameterization that result from utilizing multiple KI operators [68, 73, 82, 109, 113, 117, 125, 178]. Consequently, it is not surprising that no prior work on KDO algorithms has utilized an AOS to balance the use of KD operators created from data mining methods with the use of KI operators.

The lack of existing literature on knowledge-intensive MOEAs and KDO algorithms using an AOS prompts three research questions that are addressed in this thesis.

1. What are appropriate credit assignment strategies for an AOS on an multi-objective optimization problem?
2. Does employing knowledge-dependent operators with an AOS improve an

MOEA's search performance?

3. Does an AOS improve the search performance of KDO algorithms?

The first question explores a fundamental component of AOS strategies. Recall that an AOS monitors each operator's performance or its ability to create improving solutions through a credit assignment strategy. The AOS then uses the information about the operators' performances to select the most appropriate operator at each iteration. Much work has been conducted on how an AOS selects the next operator once the operator performance metrics are given, and these operator selection strategies have been empirically compared in several comparative experiments [58, 68, 103, 109, 174]. In contrast, there is no existing study comparing the credit assignment strategies for an AOS on a multiobjective optimization problem [68,73,82,109,113,117,125,178]. The absence of such a comparative experiment provides little guidance for algorithm developers on which credit assignment strategy to use when implementing an AOS. Since the proposed approach for a knowledge-intensive MOEA utilizes an AOS, a better understanding of the credit assignment strategies is required.

The second question addresses the benefits of utilizing KD operators with an AOS. Given the same amount of computational resources (e.g. number of function evaluations or wall clock time), a successful knowledge-intensive MOEA should be able to discover a higher-quality set of solutions than a knowledge-independent MOEA even when provided with multiple KD operators that might contain conflicting information. Moreover, to elucidate the advantages and disadvantages of the proposed method, the use of KD operators should be compared to the use of KD constraints, which are also used in existing algorithms. There is no prior work that examines the performance of an MOEA incorporating knowledge through operators versus through constraints, so it is unclear when one approach is more appropriate than the other.

The final question examines the application of an AOS within a KDO algorithm to adaptively apply KD operators that are created during the optimization. As with a successful knowledge-intensive MOEA, a successful KDO algorithm should be able discover a higher-quality set of solutions than a conventional MOEA when given the same amount of computational resources. No prior work has attempted to extend a KDO algorithm with an AOS, and therefore, an AOS's contribution to the KDO search strategy must be examined carefully. Specifically, the application of the knowledge-dependent operators by an AOS should be compared with existing methods that are less adaptive.

## 1.6 Structure

The structure of this thesis is as follows.

Chapter 2 provides a survey of the literature on existing credit assignment strategies for AOS on multiobjective problems. Based on the existing literature, a classification of these strategies is developed for the first time, identifying nine categories based on the type of fitness function and set of solutions used to assess an operator's impact. This classification reveals gaps in the literature and five new credit assignment strategies are proposed to fill the gaps. This chapter also addresses the lack of any previous comparative experiment analyzing the efficacy of existing credit assignment strategies. Nine credit assignment strategies are compared experimentally on standard benchmarking problems to evaluate their effect in elevating the generality of an MOEA and outperforming a random operator selector.

Chapter 3 develops a knowledge-intensive MOEA that utilizes an AOS to control the use of KD operators alongside KI operators. The chapter also addresses the lack of comparative experiments between knowledge-intensive EAs that apply the available knowledge through KD operators and those that utilize KD constraints. This chapter benchmarks one EA using KD operators and two EAs using KD constraints against an analogous knowledge-independent EA on a design problem for a climate-monitoring DSM. Each EA is evaluated for its ability to attain high-quality solutions with the fewest possible number of function evaluations. In addition, each method is assessed for its ability to focus on applying the knowledge that improves the search performance, handle conflicting information that suggests improving solutions with opposing modifications, and balance the exploitation of the available knowledge with the exploration of the tradespace to prevent premature convergence on local optima.

Chapter 4 introduces KDO\AOS, a novel KDO algorithm that utilizes an AOS to apply KD operators that are created from information extracted from data mining previously discovered solutions. KDO\AOS uses both KI operators such as crossover and KD operators created during the optimization by reallocating computational resources to the most effective search strategy. The efficacy of KDO\AOS is demonstrated on a multiobjective design problem for a climate monitoring DSM and benchmarked against other KDO algorithms that apply the extracted knowledge through less adaptive methods.

Chapter 5 provides a summary of the thesis and its contributions to the literature. Limitations of the proposed work are addressed and opportunities for future work are discussed.

CHAPTER 2

# A CLASSIFICATION AND COMPARISON OF CREDIT ASSIGNMENT STRATEGIES FOR MULTIOBJECTIVE ADAPTIVE OPERATOR SELECTION

## 2.1 Introduction

Multiobjective evolutionary algorithms (MOEA) are popular methods when solving a real-world problem because they can handle multiobjective, multi-modal, nonconvex, and nonlinear problems. Their popularity has led to a prolific area of research producing many MOEAs such as NSGA-II [39], SPEA2 [201], IBEA [202], and MOEA/D [193], but the success of a given MOEA is largely dependent on its parameterization and on the problem it is solving [72, 97]. With the myriad of MOEAs in the literature, the possibly infinite ways to parameterize them, and little to no *a priori* information about the search landscape of the problem, a user has difficulty finding the best algorithm and parameters to solve the problem at hand. Furthermore, in real-world applications, the problem formulation occasionally changes as the decision makers or algorithm users learn more about the problem and change the range or number of decision variables, add or change the objectives, or refine the model used to evaluate the solutions. This dynamic problem formulation causes even more distress to the algorithm user, because the algorithm's parameters, in general, must be re-tuned for the current problem formulation [48]. While parameter tuning methods exist, they can be expensive and time consuming [48,59], making them impractical for some real-world applications. Therefore, it is desirable to have an optimization method whose performance is robust to the problem formulation such that more effort can be spent on refining the problem formulation and on analysis rather than on algorithm development or parameter tuning.

Adaptive operator selection (AOS) is one approach to increase the generality of heuristic optimization methods such as MOEAs [32, 59, 97, 122]. An AOS is a high-level controller that searches the decision space by selecting and applying an operator at every iteration based on its past performance and rewarding or penalizing it for the solutions it produces. An AOS strategy is defined by its two main components: the *credit assignment* strategy to evaluate the performance of an operator with a scalar metric and the *operator selection* method to select the next operator to apply based on its performance [32]. While there are many existing AOS strategies for single-objective problems [10, 17, 19, 30, 32, 44, 57, 58, 70, 103, 120–123, 151, 170], there are comparatively few for multiobjective problems (MOP) [68, 69, 73, 82, 108, 109, 113, 117, 125] despite the plethora of research on MOEAs. Unfortunately, migrating existing AOS strategies from single-objective problems to MOPs is nontrivial because defining credit assignment strategies for MOPs is neither straightforward nor well understood. Moreover, there is no known comparative study examining the efficacy of existing credit assignment strategies for MOPs, so it is unclear which strategies are most appropriate.

The purpose of this chapter is to provide greater insights and recommendations on multiobjective credit assignment strategies, which we hope will spur more work on AOS for MOPs. The main contributions of this chapter are to 1) introduce a classification of credit assignment strategies, 2) introduce new credit assignment strategies based on research gaps revealed by the classification, and 3) conduct a comparative experiment of nine different credit assignment strategies to identify effective approaches. To the best of our knowledge, there is currently no classification available for multiobjective credit assignment strategies to offer guidance when defining new strategies. The proposed classification categorizes credit assignment strategies by the MOEA method to compute the fitness of solutions (e.g. decomposition- , indicator-, dominance-based) and the different sets of solutions used to compute the credits (e.g. parent solution, Pareto front, neighborhood). This classification reveals research gaps in credit assignment strategies for MOPs, and helps us define new strategies to fill the gaps. Finally, this paper offers recommendations on credit assignment strategies for MOPs through comparative experiments. To date, comparative experiments with AOS have only examined the efficacy of different credit assignment strategies for single-objective problems [57, 70, 121] or operator selection methods [58, 68, 103, 109, 174].

Unlike credit assignment strategies, operator selection methods can be easily extended to MOPs because the credit assignment strategies measure an operator's performance with a scalar value, so there is no difference between single-objective and multiobjective problems for the operator selector. Therefore, this paper seeks to address the research gap of credit assignment strategies for MOPs and does not investigate the different operator selectors. However, this paper provides insights on the dynamics of an operator's credits to complement studies examining the effects of fast and slow dynamics on an operator selector's performance [58, 174]. The combined knowledge will help users select an effective combination of credit assignment strategy and operator selector such that the overall AOS will perform well over a range of problems.

The remainder of this paper is organized as follows. Section 2.2 provides a background on MOEAs and AOS. Section 2.3 presents the classification of credit assignment strategies and provides a detailed overview of existing credit assignment strategies for multiobjective AOS. The research gaps identified by the classification are filled in by newly proposed credit assignment strategies. The experimental setup and results of the comparative study are presented in Section 2.4 and Section 2.5, respectively. Section 2.6 summarizes findings on the different credit assignment strategies and Section 2.7 discusses the conclusions and future work.

## 2.2 Background

### 2.2.1 Multiobjective Evolutionary Algorithms

A critical component to the success of an MOEA is its fitness function because it determines which individuals will remain in the population. Defining a fitness function in single-objective problems is relatively straightforward because an individual's fitness is typically a simple function of its objective value. For MOPs, however, it is more complicated because the fitness function must map the vector of objective values to a scalar value. Moreover, the fitness function of an MOEA must simultaneously promote the progress of the approximate Pareto front $PF$ toward the true Pareto front $PF^*$ and the diversity of the solutions in $PF$.

Although a vast amount of work explores different fitness functions, MOEAs can be categorized into three main groups based on their fitness function [111,112, 179]: dominance-based, indicator-based, and decomposition-based. In dominance-based MOEAs, if $x \prec y$ then $x$ is preferred over $y$, and pairwise comparisons of the individuals establish a partial ordering of the individuals such as Pareto ranking in NSGA-II [39] or Pareto strength in SPEA2 [201]. It is common for dominance-based MOEAs to use a density estimator such as crowding distance as a secondary selection criterion to establish a total order. Indicator-based MOEAs such as IBEA [202] and its variants use a quality indicator (e.g. hypervolume or R2 indicator) to assign a fitness to solutions. For each individual in the population, IBEA and R2-IBEA [148] use a binary indicator to compare two solutions and assign an individual's fitness by aggregating the indicator values of pairwise comparisons to the rest of the individuals in the population. Finally, decomposition-based MOEAs decompose the $m$-dimensional MOP into $k$ single objective subproblems, often using a Tchebycheff approach like in MOEA/D [193, 194]. The fitness of an individual is related to its ability to optimize one or more of the subproblems.

While MOEAs have been hugely popular, there are three concerns. Firstly, MOEAs have various parameters that generally need to be tuned for a specific problem in order to efficiently and effectively approximate $PF^*$. Hadka et al. performed a comprehensive sensitivity analysis of 9 MOEAs on 33 benchmarking problems by testing thousands of parameter configurations for each MOEA on each problem [72]. Their results showed that the performance of many MOEAs, including NSGA-II and IBEA, is sensitive to their parameterization. Secondly, choosing the MOEA operators amongst the myriad provided in the literature is not straightforward [17] because some operators are effective on problems with specific properties such as linear separability [84], and the operators also have parameters that may need to be tuned. This leads to a difficult and time consuming process of not only finding the right combination of an MOEA and operators, but also tuning their parameters [17]. Finally, an optimal parameterization at the beginning of the search may be suboptimal by the end of the search [48, 57]. To this extent, some adaptive MOEAs encode parameters in an individual's chromosome in an attempt to adjust the parameters during the search process [48, 97], but combining the parameters with the decisions variables in the same chromosome increases the size of the problem, which can make the problem harder to solve.

### 2.2.2 Adaptive Operator Selection (AOS)

AOS is one approach to combat the difficulty in deciding on the appropriate operators and parameters to solve the problem at hand [97]. AOS is related to hyperheuristics [17] consisting of a high-level controller that adaptively selects high performing operators from a candidate set of recombination or mutation operators. Note that hyperheuristics or multimethod algorithms such as AMALGAM [178] or HHMO_CF [117] differ from AOS primarily in the selection of metaheuristics (e.g. NSGA-II, differential evolution [171]) as opposed to operators and are otherwise similar. Therefore, we reference work on multi-method algorithms in addition to those on AOS strategies.

The general flow of an MOEA with an AOS strategy follows Fig. 2.1. At iteration $t$, the AOS selects an operator to generate offspring solutions. The offspring are evaluated and inserted into the population and updated according to the underlying MOEA. If the operator has a positive impact on the search process, it is rewarded credits, which increases its chance to be selected by the AOS in successive iterations. The search continues until a termination criterion is met.

The two main components of an AOS strategy are the *credit assignment* strategy that defines how to reward an operator based on its impact in the search process and the *operator selection* strategy that uses the rewards to determine the next operator to apply [32]. Both are detailed in the following subsections. For brevity, AOS strategies and credit assignment strategies will be referred to as AOS and credit assignments, respectively, for the remainder of this paper.

### Credit Assignment

The most common credit assignment rewards an operator for its ability to improve the fitness of an offspring solution over its parent solution(s) [57]. Other credit assignments also reward the operators that generate the ancestors, beyond the parent solution, of a high quality offspring [30], but it is unclear if such credit assignments are effective [55]. Fialho et al. suggest that more credit should be rewarded to operators that make rare but large improvements as opposed to operators that make frequent, small improvements [56, 57]. Finally, some credit assignments, such as Compass [123], reward operators that can improve the fitness of a solution while maintaining diversity in the solution set.

Research on credit assignments focuses on AOS for single-objective problems [30, 57, 121], and there is a lack of credit assignments for AOS to solve MOPs [68, 109]. The main challenge of developing an AOS for MOPs is in defining an appropriate credit assignment because it is not obvious how one should compare a solution with another, especially when there are a multitude of existing MOEA fitness functions. Furthermore, many single-objective AOS use single-point search [17], but with population-based search methods such as MOEAs, it is not clear which individuals in the population should be analyzed in every iteration.

The existing credit assignments for MOPs differ in which individuals are compared and how they are compared, but with no comparative study examining their efficacy, it is difficult to assess which credit assignments are most appropriate for MOPs. MOSaDE [82], which uses differential evolution, rewards an operator if it successfully replaces an incumbent solution using Pareto-dominance and crowding distance as a secondary measure. In MCHH [125], an operator is rewarded for creating offspring that dominate many solutions from the previous

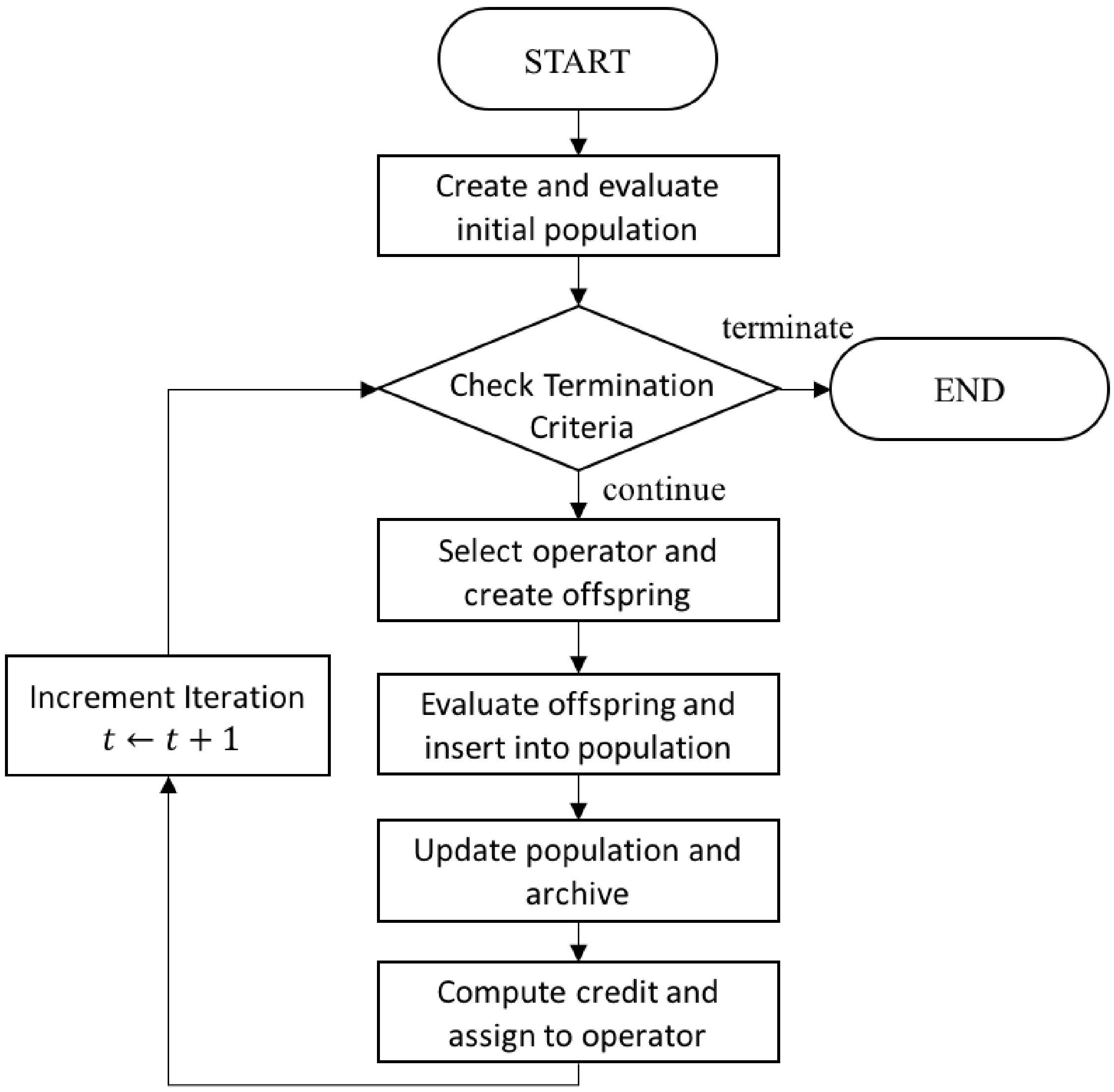
START
Create and evaluate initial population
Check Termination Criteria
terminate
END
continue
Select operator and create offspring
Evaluate offspring and insert into population
Update population and archive
Compute credit and assign to operator
Increment Iteration $t \leftarrow t + 1$

Figure 2.1: The general flow of an AOS strategy for MOEAs

generation. Adap-MODE [108] rewards an operator for improving the fitness of a solution, where fitness is a weighted sum of a Pareto strength metric and a density metric. In more current work, Li et al. develop MOEA/D-FRRMAB [109], which uses a decomposition-based fitness function to reward an operator for its ability to improve the objective value of a subproblem and its neighboring subproblems. Borg [73], on the other hand, uses dominance-based fitness to reward an operator for the number of solutions it has contributed to the external archive.

Similarly, after updating its population using the nondominated sorting procedure from NSGA-II [39], AMALGAM [178] rewards a metaheuristic for the number of solutions it contributes to the population. Preliminary work by the authors examined three different credit assignments that also use dominance-based fitness but instead reward operators for creating offspring that dominates their parent (ODP-IC), inserting offspring in the Pareto front (PF-IC), or for its contribution to the current Pareto front (PF-AC) [76]. Finally, HHMO_CF [117] uses the D-metric [199], an extension of the hypervolume indicator, to reward an MOEA for improving the quality of the population over the previous generation.

### Operator Selection

Using the assigned credits, an AOS selects an operator at every iteration, but is faced with the exploration versus exploitation dilemma commonly seen in search algorithms. While it is advantageous to select operators with many credits, it is also beneficial to occasionally explore the poorly performing operators because they may begin to produce high quality solutions as the population evolves.

A common operator selection strategy is probability matching (PM) [67]. Given a finite set of operators $O$, an operator $o_i \in O$ is selected at iteration $t$ with a probability $p_{i,t}$, proportional to the operator's quality $q_{i,t}$, which is determined from the credits $c_{i,t}$ received by $o_i$. A minimum selection probability, $p_{min} > 0$, is defined to foster the exploration of poorly performing operators. The quality update rule for an operator $o_i$ that receives credit $c_{i,t}$ is shown in (2.1), where the adaptation rate $\alpha \in [0, 1]$ controls the importance of recently received credits compared to older credits received in the past. The update rule for selection probabilities is shown in (2.2), where $p_{min} \leq \frac{1}{|O|}$ and $q_{i,t} \geq 0$ to guarantee $p_{i,t} \in [0,1]$.

$$q_{i,t+1} = (1-\alpha) \cdot q_{i,t} + \alpha \cdot c_{i,t} \tag{2.1}$$

$$p_{i,t+1} = p_{min} + (1 - |O| \cdot p_{min}) \cdot \frac{q_{i,t+1}}{\sum_{j=1}^{|O|} q_{j,t+1}} \tag{2.2}$$

Thierens argues that PM does not perform well when there are several mediocre operators and only a few high performing operators [174]. The selection probabilities from many mediocre performing operators can sum to a significant value with respect to that of the high performing operators, and PM can have difficulty selecting the high performing operators amongst the mediocre ones. Adaptive pursuit (AP) tries to address the shortcomings of PM by employing a greedier selection strategy [174]. Following the same principles in PM, AP also utilizes $p_{min}$ to foster exploration, but AP uses a different probability update rule. AP first identifies the operator with the greatest quality, $o^*$, and defines the probability, $p_{max}$, as shown in (2.3) and (2.4). Then, AP begins to asymptotically pursue $o^*$ with $p_{max}$ and all others with $p_{min}$ by updating the probabilities with (2.5) where $\beta \in [0,1]$ is the learning rate.

$$o^* = \underset{o_i \in O}{\operatorname{argmax}}\, q_{i,t} \tag{2.3}$$

$$p_{max} = (1 - (|O| - 1) \cdot p_{min}) \tag{2.4}$$

$$p_{i,t+1} = \begin{cases} p_{i,t} + \beta \cdot (p_{max} - p_{i,t}) & \text{if } o_i = o^* \\ p_{i,t} + \beta \cdot (p_{min} - p_{i,t}) & \text{otherwise} \end{cases} \tag{2.5}$$

Balancing exploitation and exploration of the operators is crucial because the performance of the operators, and therefore the distribution of the credits they receive, is dynamic. During the search, operators may experience small or large fluctuations in their performance as the population evolves, and the dynamics in the distribution of the credits received by the operators can greatly affect the efficacy of an operator selector [58]. Fialho et al. show that PM and AP are not effective when the dynamics in the credits are fast because they cannot adapt their selection policy quick enough. Other operator selectors such as dynamic multi-armed bandit (DMAB) [32] or sliding multi-armed bandit (SlMAB) [58] are able to adapt to faster dynamics by tuning their hyper-parameters.

Nonetheless, this paper only experiments with PM and AP for their simplicity and intuitive parameters. Other operator selection strategies based on multi-armed bandit algorithms [32, 58, 59, 103], choice functions [30, 69, 117], dynamic island models [19, 170], or Markov chain models [125] have more hyper-parameters that can affect the behavior of the AOS [58]. In addition, the purpose of the paper is not to find the best operator selector and credit assignment combination, but rather to experiment with different credit assignments for MOPs.

## 2.3 Multiobjective Credit Assignment

Existing multiobjective credit assignments, discussed earlier, differ in 1) the solutions or solution sets used as inputs (e.g. offspring and parent, offspring and Pareto front) and 2) the MOEA fitness function (i.e. decomposition-, indicator-, or dominance-based) used to compare the input solutions. A credit assignment can be defined by a function $h(\phi, \psi, o_i) = c_{i,t}$, which computes the credit $c_{i,t}$ assigned to an operator $o_i$ at iteration $t$ with inputs solution sets $\phi$ and $\psi$. Using this formulation and the two defining dimensions above, we propose a classification of multiobjective credit assignments shown in Table 2.1. The rows of the table show three groupings for the inputs to $h(\phi, \psi, o_i)$ called offspring vs. parent (OP), set improvement (SI), and contribution to the set (CS), while the columns of the

Table 2.1: A classification of multi-objective credit assignments

| | | MOEA Fitness Function | | |
|---|---|---|---|---|
| | | Decomposition | Dominance | Indicator |
| Inputs | OP | | Adap-MODE [108]<br>MOSaDE [82]<br>ODP-IC [76] | |
| | SI | MOEA/D-UCB [68]<br>MOEA/D-FRRMAB [109]<br>MOEA/D-CDE [113] | MCHH [125]<br>PF-IC [76] | MOHH_CF [117] |
| | CS | | Borg [73]<br>AMALGAM [178]<br>PF-AC [76] | |

table show the three types of MOEA fitness functions. The empty cells in Table 2.1 reveal that several credit assignment classes have not yet been explored. The following subsections describe each of the nine multiobjective credit assignment categories and propose new credit assignments to fill the empty cells of Table 2.1.

### 2.3.1 Inputs to Credit Assignment Function $h(\phi, \psi, o_i)$

As stated earlier, in population-based searches, it is not straightforward which solutions or solution sets should be used to assess an operator's impact. Based on the existing multiobjective credit assignments in Table 2.1, we propose three categories of inputs $\phi$ and $\psi$.

The first category is called *offspring vs. parent* (OP) and takes the form $h_{OP}(x^p, x^{o_i,t}, o_i)$, where $x^p$ is a parent solution and $x^{o_i,t}$ is the offspring solution created at iteration $t$ by operator $o_i$. OP credit assignments reward an operator based on the fitness improvement of $x^{o_i,t}$ over $x^p$, and only the operator applied at time $t$ is rewarded. This follows credit assignments commonly seen in single-objective AOS [57, 58], and it is computationally inexpensive involving only one pairwise comparison. However, OP only provides information about an operator's local impact and does not account for its overall impact to the population. In this paper, if $x^{o_i,t}$ has more than one parent, such as in crossover operations, a random parent is assigned as $x^p$ with uniform probability.

The second category of credit assignments is called *set improvement* (SI), and compared to OP, SI provides a broader perspective of an operator's impact. The SI function $h_{SI}(S, x^{o_i,t}, o_i)$ compares $S$, the objective vectors of a set of solutions, with the union set $S \cup x^{o_i,t}$, where $x^{o_i,t} \notin S$, and it rewards operators creating offspring that improve $S$ by increasing its diversity or pushing convergence of $S$ to $PF^*$. Candidates for $S$ are the current population, the neighborhood of $x^{o_i,t}$, the current Pareto front $PF$, or the archive. SI only rewards the operator applied at time $t$ and will favor operators that make frequent improvements to $S$.

Finally, the third category of credit assignments is called *contribution to set* (CS), which takes the most holistic view by rewarding an operator for its total contribution to $S$ instead of just the contribution of its most recent offspring. The CS function $h_{CS}(S, , o_i)$ computes the contribution of $o_i$ to $S$, and is used in Borg [73] to reward $o_i$ for the number of solutions it has contributed to the archive. Candidates for $S$ are the current population, the neighborhood of $x^{o_i,t}$, the current Pareto front $PF$, or the archive. Since the composition of $S$ can change significantly with the insertion of even a single solution, the contribution of each operator must be recomputed every iteration. Therefore, as opposed to OP and SI, CS computes the performance of all operators and rewards them at every iteration, even those not applied at $t$. CS credit assignments will favor operators that make large improvements to $S$ as opposed to those that make incremental improvements to $S$. Solutions that make large improvements to $S$ will remain in $S$ for several iterations, and an operator will continue to receive credit for them. A solution that makes a small improvement to $S$, however, may be quickly replaced by another solution.

### 2.3.2 MOEA Fitness Function in $h(\phi, \psi, o_i)$

Each credit assignment function, $h(\phi, \psi, o_i)$, requires a method to compute the fitness of a solution in order to compare the input solutions or sets of solutions from the previous subsection. We turn to MOEA fitness functions to define the OP, SI and CS credit assignment functions, and the following subsections are divided into decomposition-based, indicator-based, and dominance-based multiobjective credit assignments. Definitions of existing and newly proposed credit assignments are provided in their respective subsections. In addition, the computational complexity of each credit assignment used in our comparative study is provided in terms of $m$ the number of objectives, $n$ the number of solutions in the population, $n_{PF}$ the number of solutions in the Pareto front, $n_T$ the size of the neighborhood $T$ of a solution, and $\Lambda$ the set of weight vectors to compute the R2 indicator value.

Table 2.2: Summary of multiobjective credit assignments

| Type | MOEA Fitness | Abbreviation | Rewards $o_i$: | Complexity |
|---|---|---|---|---|
| OP $h_{OP}(x^p, x^{o_i,t}, o_i)$ offspring vs. parent | Decomposition | **OP-De** | if $x^{o_i,t}$ improves the current subproblem | $O(m)$ |
| | Dominance | OP-Do | if $x^{o_i,t} \prec x^p$ | $O(m)$ |
| | Indicator | **OP-I** | if binary R2 indicator $I_{R2}(x^p, x^{o_i,t}) > 0$ | $O(m \cdot \lvert\Lambda\rvert)$ |
| SI $h_{SI}(S, x^{o_i,t}, o_i)$ set improvement | Decomposition | SI-De | if $x^{o_i,t}$ improves the current or neighboring subproblems | $O(m \cdot n_T)$ |
| | Dominance | SI-Do | if $x^{o_i,t}$ enters $PF$ | $O(m \cdot n_{PF})$ |
| | Indicator | **SI-I** | if $x^{o_i,t}$ improves R2 indicator of $P$ | $O(m \cdot \lvert\Lambda\rvert \cdot n)$ |
| CS $h_{CS}(S, , o_i)$ contribution to set | Decomposition | **CS-De** | for contribution of solutions to neighborhood | $O(m \cdot n_T)$ |
| | Dominance | CS-Do | for contribution of solutions to $PF$ | $O(m \cdot n_{PF})$ |
| | Indicator | **CS-I** | for solutions in $P$ that contribute to R2 indicator | $O(m \cdot \lvert\Lambda\rvert \cdot n)$ |

There are too many possible definitions for multiobjective credit assignments to present a comprehensive list in this paper, so we only give one example definition for each category but provide suggestions for others at the end of this section. Table 2.2 shows a summary of the credit assignments used in this paper with their associated type, abbreviated name, and computational complexity. Names in bold indicate the new credit assignments introduced in this paper.

### Decomposition-based Credit Assignment

A natural way to transition AOS from single-objective problems to MOPs is to use a decomposition-based credit assignment [109]. The $m$-objective problem is decomposed into $k$ single-objective problems using an algorithm such as MOEA/D, and an operator is rewarded an amount proportional to the improvement in the objective value of an offspring solution to that of its parent. MOEA/D creates single-objective subproblems using a set of $k$ weight vectors $V = \{v^1, \ldots, v^k\}$, where $1 = \sum_{i=1}^{m} v_i^j$ and $v_i^j \geq 0$ for all $i = \{1, \ldots, m\}$ and for all $j = \{1, \ldots, k\}$. The objective for a subproblem $v^j$ is to minimize the weighted Tchebycheff distance in the objective space between the subproblem's solution $x$ and the utopia point $u^*$, as shown in (2.6). If $n_T$ is the size of the neighborhood, each subproblem has a neighborhood $T$ that includes the $n_T$ closest weight vectors using Euclidean distance. A new offspring is produced for $v^j$ by combining parents solutions from $v^j$ and from its neighboring subproblems, and the offspring replaces up to a maximum number $n_R$ of incumbent solutions for both $v^j$ and its neighboring subproblems if it minimizes the objective value for that subproblem. For further detail see [193,194].

$$g^{te}(x|v^j, u^*) = \max_{1 \leq i \leq m} \{v_i^j |x - u_i^*|\} \tag{2.6}$$

*OP-De*: The OP-type decomposition-based credit assignment is referred to as OP-De. It compares the objective value of $x^{o_i,t}$ with that of $x^p$ on subproblem $v^j$ and is given by (2.7). $o_i$ receives credit only if $x^{o_i,t}$ replaces $x^p$. The runtime to compute OP-De is $O(m)$.

$$h_{OP\text{-}De}(x^p, x^{o_i,t}, o_i) = \frac{g^{te}(x^p|v^j, u^*) - g^{te}(x^{o_i,t}|v^j, u^*)}{g^{te}(x^p|v^j, u^*)} \tag{2.7}$$

*SI-De*: SI-De is the SI-type decomposition-based credit assignment used by FRRMAB-MOEA/D [109], MOEA/D-UCB [68], and MOEA/D-CDE [113]. SI-De rewards $o_i$ for its ability to improve the objective value of not only subproblem $v^j$, but also the subproblems in the neighborhood of $v^j$. SI-DE uses (2.8) to reward $o_i$, where $R$ is the set of incumbent objective vectors replaced by $x^{o_i,t}$. Since it is possible that $x^{o_i,t}$ may be checked against all solutions in $T$ before $n_R$ solutions are replaced, the complexity to compute SI-De is $O(m \cdot n_T)$.

$$h_{SI\text{-}De}(R, x^{o_i,t}, o_i) = \sum_{y \in R} h_{OP-De}(y, x^{o_i,t}, o_i) \tag{2.8}$$

*CS-De*: The CS-type decomposition-based credit assignment, CS-De given in (2.9), measures the contribution of $o_i$ by counting how many solutions it has created in $T^j$, which is the set of incumbent solutions in the neighborhood of subproblem $v^j$. The awarded credit is normalized by $n_T = |T^j|$. Every solution is tagged with its operator identification to keep track of which operator created a given solution. Updating $T^j$ with $x^{o_i,t}$ and computing CS-De has complexity $O(m \cdot n_T)$.

$$h_{CS\text{-}De}(T^j, , o_i) = \frac{1}{n_T} \sum_{x^{o_i,t} \in T^j} 1 \tag{2.9}$$

### Dominance-based Credit Assignment

Pareto dominance is a popular choice for MOEA fitness functions, explaining why most of the existing multiobjective credit assignments use Pareto dominance relationships to measure an operator's performance. Like many dominance-based MOEAs, some dominance-based credit assignments also use a density metric such as crowding distance in NSGA-II [39]. For simplicity, we do not implement density metrics in the credit assignments presented in this paper, but discuss potential implementations in the end of this section.

*OP-Do*: Dominance-based OP-type credit assignment have previously been used in MOSaDE [82] and Adap-MODE [108]. In MOSaDE, which uses differential evolution, an operator is rewarded if the offspring dominates the parent solution or has a larger crowding distance if the two solutions are nondominated with respect to each other. Adap-MODE first assigns a fitness value to all individuals using a Pareto strength metric combined with a density metric and then rewards an operator for the improvement in the offspring fitness over its parents. Similar to MOSaDE, ODP-CI [76] is a credit assignment defined in previous work that rewards an operator if it creates an offspring $x^{o_i,t}$ that dominates its parent $x^p$. We reuse ODP-CI in this paper but refer to it as OP-Do. OP-Do defined in (2.10) has a binary outcome that only provides information on whether or not $x^{o_i,t}$ is preferred over $x^p$ and not by how much. OP-Do has a runtime of $O(m)$.

$$h_{OP\text{-}Do}(x^p, x^{o_i,t}, o_i) = \begin{cases} 1 & \text{if } x^{o_i,t} \prec x^p \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

*SI-Do*: The credit assignment in MCHH [125] and PF-IC [76] take a broader perspective of an operator's impact by implementing an SI-type credit assignment.

In MCHH, an operator is rewarded if it creates offspring that dominate many solutions from the previous generation. PF-IC, previously introduced by authors, rewards an operator $o_i$ if it creates $x^{o_i,t}$ such that it enters the current Pareto front $PF$ because a solution entering $PF$ improves it by either pushing convergence towards $PF^*$ by replacing one or more existing solutions in the set or increasing the diversity if it doesn't replace any existing solutions in $PF$. In this paper, we reuse PF-IC but rename it as SI-Do. SI-Do is given in (2.11), and since $x^{o_i,t}$ must be compared with all solutions in $PF$, the runtime for SI-Do is $O(m \cdot n_{PF})$. Similar to OP-Do, SI-Do also results in a binary outcome; $x^{o_i,t}$ either enters or does not enter $PF$.

$$h_{SI\text{-}Do}(PF, x^{o_i,t}, o_i) = \begin{cases} 1 & \text{if } x^{o_i,t} \text{ enters } PF \\ 0 & \text{otherwise} \end{cases} \tag{2.11}$$

*CS-Do*: Borg [73] uses a dominance-based CS-type credit assignment where an operator is rewarded an amount proportional to the number of solutions it has created in the archive. Similarly, PF-CA, previously introduced by the authors [76], examines the solutions in $PF$ instead of the archive. The credit assignment function used in this paper is analogous to that of Borg and PF-CA, and will be referred to as CS-Do. It is defined in (2.12), and as in CS-De, every solution is tagged with its operator identification to track which operator created a given solution. To maintain and check $PF$, CS-Do has a runtime of $O(m \cdot n_{PF})$.

$$h_{CS-Do}(PF, , o_i) = \frac{1}{n_{PF}} \sum_{x^{o_i,t} \in PF} 1 \tag{2.12}$$

**Indicator-based Credit Assignment**

To the best of the authors' knowledge, indicators have not yet been used in AOS credit assignments despite their popular use for solving multiobjective problems

[11, 16, 43, 111, 112, 148]. As previously mentioned, however, an indicator-based credit assignment is used in the hyperheuristic HHMO_CF [117], which uses the D-metric [199] to reward an MOEA for improving the quality of the population over the previous generation. Inspired by HHMO_CF, we propose new indicator-based credit assignments for multiobjective AOS.

Several indicators are used in indicator-based MOEAs including the additive epsilon indicator [202], the R2 indicator [16,43,148], and the hypervolume indicator [11, 202]. We use the R2 indicator instead of the popular hypervolume indicator because it provides a similar measure to the hypervolume indicator but is less computationally expensive [16].

The R2 indicator with the standard weighted Tchebycheff function is shown in (2.13), where $S$ is a solution set, $z^*$ is a reference point, and $\Lambda$ is a set of weight vectors. For our experiments, we use 50 and 91 uniformly distributed vectors for two and three-dimensional problems, respectively, which are obtained from jMetal [46], an open source Java platform for creating and testing MOEAs. The small number of weight vectors provides an approximation of the R2 value without making the indicator calculation computationally prohibiting.

$$R2(S, \Lambda, z^*) = \frac{1}{|\Lambda|} \sum_{\lambda \in \Lambda} \min_{x \in S} \{ \max_{1 \leq j \leq m} \lambda_j \cdot |z^*_j - x_j| \} \tag{2.13}$$

*OP-I*: R2-IBEA [148] provides a binary R2 indicator, $I_{R2}(x, y)$, to compare two solutions, so it is directly applicable to OP-type credit assignments. $I_{R2}(x, y)$ is dominance-preserving, so for solutions $x$ and $y$, $I_{R2}(x, y) \leq I_{R2}(y, x)$ if $x \prec y$. The binary R2 indicator from R2-IBEA is shown in (2.14) and is used to define OP-I shown in (2.15). If the indicator value is negative, zero credit is awarded. Since

OP-I only compares one pair of solutions, it has complexity $O(m \cdot |\Lambda|)$.

$$I_{R2}(x, y) = R2(x, \Lambda, z^*) - R2(x \cup y, \Lambda, z^*) \tag{2.14}$$

$$h_{OP\text{-}I}(x^p, x^{o_i,t}, o_i) = \max\{0, I_{R2}(x^p, x^{o_i,t})\} \tag{2.15}$$

*SI-I*: Recent MOEAs such as SMS-EMOA [11], R2-MOEA [43], and R2-EMOA [16] use a solution's contribution to an indicator value as the fitness value. The same concept is used for SI-type indicator-based credit assignments, where the contribution of a solution, $x \notin S$, to the R2 indicator value is computed as:

$$C_{R2}(x, S, \Lambda, z^*) = R2(S \cup x, \Lambda, z^*) - R2(S, \Lambda, z^*) \tag{2.16}$$

In this paper, we employ the fast computation of the R2 contribution given by Diaz-Manriquez et al. [43], which has complexity $O(m \cdot |\Lambda| \cdot |S|)$. SI-I, defined in (2.17), uses $C_{R2}$ to examine if $x^{o_i,t}$ improves the indicator value of the population $P$. The absolute value converts improvements to the R2 indicator value to positive credits. SI-I has complexity $O(m \cdot |\Lambda| \cdot n)$.

$$h_{SI\text{-}I}(P, x^{o_i,t}, o_i) = |C_{R2}(x^{o_i,t}, P, \Lambda, z^*)| \tag{2.17}$$

*CS-I*: CS-I, defined in (2.19), also uses $C_{R2}$. For each operator $o_i$, CS-I computes the contribution $C_i$ shown in (2.18), which is the sum of the absolute $C_{R2}$ values belonging to the solutions in the population $P$ created by $o_i$. $C_i$ is then normalized by $C_{min}$ and $C_{max}$, which are the minimum and maximum $C_i$ achieved by any operator. Using the fast computation of the R2 contribution, finding $C_{R2}$ for all $x \in P$ has complexity $O(m \cdot |\Lambda| \cdot n)$. Therefore, CS-I has complexity $O(m \cdot |\Lambda| \cdot n)$.

$$C_i = \sum_{x^{o_i,t} \in P} |C_{R2}(x^{o_i,t}, P, \Lambda, z^*)| \tag{2.18}$$

$$h_{CS\text{-}I}(P, x^{o_i,t}, o_i) = \frac{C_i - C_{min}}{C_{max} - C_{min}} \tag{2.19}$$

**Alternative Credit Assignments**

While this section presented only nine multiobjective credit assignments, there are many other possible definitions. Dominance-based credit assignments could benefit from the use of density information when solving many-objective problems with more than three objectives where information gained from Pareto dominance relations diminishes [110–112,150,179]. It has been shown that modified density estimates can make dominance-based MOEAs competitive with decomposition-based or indicator-based MOEAs [110], so it may also be useful in credit assignments. Furthermore, modified Pareto dominance relationships such as $\epsilon$-dominance [105] may be used in place of the regular dominance relationship. MOEAs using $\epsilon$-dominance [38, 73] have also been successful in solving many-objective problems. Lastly, different indicators, such as the hypervolume indicator can be used for the indicator-based credit assignments.

It should also be noted that one can combine categories in Table 2.1 to define a credit assignment that uses a hybrid fitness function like BCE algorithms [111] that use both dominance and non-dominance based fitness functions or MOEA/DD [107] that uses dominance and decomposition-based fitness. These hybrid MOEAs among others [11, 16] have shown promise in solving many-objective problems, so credit assignments may also benefit from a similar hybrid approach.

## 2.4 Experimental Setup

Our experimental studies investigate the effect of each credit assignment on the efficacy of the AOS by combining each credit assignment with an operator selector and an MOEA for solution management tasks such as selecting solutions for the operators and updating the population and archive. We implement decomposition-based credit assignments with MOEA/D-DRA [194], dominance-based credit assignment with NSGA-II [39], and indicator-based credit assignments with IBEA [202]. These algorithms are the representative MOEAs for each method (i.e. decomposition-, dominance-, and indicator-based). The three MOEAs are modified using their respective implementations from MOEAFramework [71], another open source Java platform for creating and testing MOEAs. The source code used for this paper is publicly available at https://github.com/seaklab/mopAOS.

### 2.4.1 AOS Algorithms

#### Decomposition-based AOS

Decomposition-based credit assignments require a decomposition-based MOEA such as MOEA/D, so we follow MOEA/D-FRRMAB [109] and use MOEA/D-DRA [194], a variant of MOEA/D that adaptively focuses computational effort to subproblems that have progressed the least by assigning utilities to each subproblem, where the utility, $\pi^j$, decreases as the subproblem $v^j$ improves its objective function. The pseudocode for the MOEA/D-DRA AOS is given in Algorithm 1. For each subproblem that is selected, an offspring is generated from parent solutions selected from the mating pool, which is determined as the solutions to the neighboring subproblems with probability $\delta$, or the whole population, otherwise. An offspring solution can replace up to $n_R$ neighborhood solutions if it improves the respective subproblem's objective function. To create an offspring, the AOS selects $o_i$ and the required number of parent solutions are selected from the mating

Algorithm 1: Pseudocode for MOEA/D-DRA AOS

```
1: Initialize population $P = \{x^1, \cdots, x^n\}$;
2: Compute utopia point $u^*$;
3: Compute or load weights $V = \{v^1, \cdots, v^n\}$;
4: for j=1 to n do
5:     Create neighborhood $T^j$;
6:     $\pi^j = 1$;
7: end for
8: $t \leftarrow 0$;
9: $gen \leftarrow 0$;
10: while Termination criteria have not been satisfied do
11:     $J \leftarrow$ Select indices of the subproblems to search;
12:     for all $j \in J$ do
13:         $t++$;
14:         $o_i \leftarrow$ select operator;
15:         $M \leftarrow$ select mating pool;
16:         $\gamma \leftarrow x^j$;
17:         while $o_i$ needs more parents solutions do
18:             Randomly select $x^r$ from $M$;
19:             $\gamma \leftarrow \gamma \cup x^r$;
20:         end while
21:         $x_{o_i,t} \leftarrow o_i.operate(\gamma)$;
22:         Update $u^*$;
23:         $c = 0$;
24:         while $c < n_r$ or $M$ is not empty do
25:             Randomly select a solution $x^r$ from $M$;
26:             if $g^{te}(x^{o_i,t}|v^r, u^*) \leq g^{te}(x^{r,t}|v^r, u^*)$ then
27:                 Replace $x^r$ with $x_{o_i,t}$;
28:                 $M \leftarrow M \setminus x^r$;
29:             end if
30:         end while
31:         Compute credit $c_{i,t}$ and reward to $o_i$;
32:     end for
33:     $gen++$;
34:     if $mod(gen, 50)$ then
35:         Update utility $\pi$ for all subproblems;
36:     end if
37: end while
```

pool, with one parent coming from the current subproblem. $o_i$ then creates $x^{o_i,t}$, which is evaluated, and the impact of $o_i$ is assessed with the credit assignment. We employ the uniformly distributed weight vectors provided by jMetal [46], and MOEA/D-DRA parameters used for this paper are the same as those used in MOEA/D-FRRMAB [109] and are listed in Table 2.4.

### Dominance-based AOS

Dominance-based credits are used with NSGA-II, where the credits are computed and rewarded after each new offspring is created. The pseudocode for NSGA-II AOS is shown in Algorithm 2. Since SI-Do and CS-Do use the Pareto front $PF$ to compute the credits, $PF$ is computed at the beginning of the search and updated with every new offspring. Updating $PF$ one solution at a time has a complexity $O(m \cdot n_{PF})$. As with the standard NSGA-II, in line 18, fast non-dominated sorting and crowding distance are used to select individuals for the next generation.

### Indicator-based AOS

Indicator-based credit assignments are used with IBEA, which uses a binary indicator $I(\{x\}, \{y\})$ to compare two solutions at a time. Since the R2 indicator is used in the credit assignments, we use the binary R2 indicator within IBEA. The fitness of a solution is determined by computing indicator values for all pairs of individuals in the population and aggregated according to (2.20), where $\kappa$ is a scaling factor and $c$ is the maximum absolute indicator value. The pseudocode for IBEA AOS is shown in Algorithm 3 and parameter values are given in Table 2.4. As with the standard IBEA, selection in line 22 continues to remove the least fit individual and updates the fitness values of the remaining solutions in the population until

Algorithm 2: Pseudocode for NSGA-II AOS

```
1: Initialize population $P = \{x^1, \cdots, x^n\}$;
2: $PF \leftarrow$ compute the Pareto front
3: $t \leftarrow 0$;
4: while Termination criteria have not been satisfied do
5:    $P_O \leftarrow \emptyset$
6:    while $|P_O| < n$ do
7:       $t++$;
8:       $o_i \leftarrow$ select operator;
9:       while $o_i$ needs more parents solutions do
10:         Use binary tournament to select $x$ from $P$;
11:         $\gamma \leftarrow \gamma \cup x$;
12:      end while
13:      $x^{o_i,t} \leftarrow o_i.operate(\gamma)$;
14:      $PF.update(x^{o_i,t})$
15:      $P_O \leftarrow P_O \cup \{x^{o_i,t}\}$;
16:      Compute credit $c_{i,t}$ and reward to $o_i$;
17:   end while
18:   $P \leftarrow select(P \cup P_O)$
19: end while
```

the population returns to its original size.

$$g^{fit}(x, S) = \sum_{y \in S \setminus \{x\}} -e^{-I(y,x)/(c \cdot \kappa)} \tag{2.20}$$

In the original version of IBEA, when computing the fitness, all objective vectors in the population are normalized and the reference point to compute hypervolume is set to $2.0^m$. This is because the binary hypervolume indicator favors the center of the Pareto front, and a reference point far from the Pareto front reduces this bias [202]. Similarly, the reference point $z^*$ for R2 is set far from the normalized population to $z^* = -1.0^m$ because the R2 indicator also favors solutions near the center of the Pareto front [16]. The credit assignments will use the normalized population and the same $z^*$. To keep the computational overhead to a minimum, the normalization bounds and the reference point are only updated

Algorithm 3: Pseudocode for IBEA AOS

```
1: Initialize population $P = \{x^1, \cdots, x^n\}$;
2: $t \leftarrow 0$;
3: while Termination criteria have not been satisfied do
4:     $P_O \leftarrow \emptyset$
5:     while $|P_O| < n$ do
6:         $t++$;
7:         $o_i \leftarrow$ select operator;
8:         while $o_i$ needs more parents solutions do
9:             Use binary tournament to select $x$ from $P$;
10:            $\gamma \leftarrow \gamma \cup x$;
11:        end while
12:        $x^{o_i,t} \leftarrow o_i.operate(\gamma)$;
13:        $P_o \leftarrow P_o \cup x^{o_i,t}$
14:        Compute credit $c_{i,t}$ and reward to $o_i$;
15:    end while
16:    $P \leftarrow P \cup P_O$
17:    Update bounds for normalization;
18:    $c \leftarrow argmax_{x,y \in P} |I(x,y)|$;
19:    for all $x \in P$ do
20:        $x.fitness \leftarrow \sum_{y \in P \setminus \{x\}} -e^{-I(y,x)/(c \cdot \kappa)}$;
21:    end for
22:    $P \leftarrow select(P)$
23: end while
```

every generation.

### 2.4.2 Operators

Some AOS apply adaptive operators [82,120,151], and a recent study showed that adapting operator parameters can improve the search performance of an AOS [113]. To decouple the adaptation of the operator selector and the operator parameters, however, we do not utilize adaptive operators. Instead, in this first comparative study of multiobjective credit assignments, we turn to other popular operators. We examine simulated binary crossover (SBX) [35], DE/rand/1 (DE) [171], uniform mutation (UM), parent-centric crossover (PCX) [36], unimodal normal distribution crossover (UNDX) [144], and simplex crossover (SPX) [176]. Polynomial mutation is applied to all solutions created by the above operators, except for UM, and SBX, PCX, UNDX and SPX create 2 offspring with each application. These are the same operators available to Borg's adaptive operator selector, and we reuse them here because they cover a broad range of operators.

The SBX operator simulates single-point crossover on real-valued variables and generates offspring around the parent solutions primarily along the coordinate axes of the decision space. While SBX is commonly used in MOEAs, it is not rotationally invariant, so it does not search efficiently on nonseparable problems with strongly coupled decision variables [84].

The DE, UNDX, PCX and SPX operators, on the other hand, are rotationally invariant operators and perform well on separable problems [84]. DE creates an offspring by translating one parent in the direction of the difference vector created by two other parents. UNDX creates a search line by connecting two parents with a line and uses another parent to determine the distance the offspring will lie from the search line. PCX is similar to UNDX but generates offspring in the vicinity of the parent solutions. SPX creates a geometrical simplex with vertices at the parent solutions, slightly expands the simplex, and generates offspring uniformly within the expanded simplex.

While the other operators require more than one parent solution, UM perturbs the decision variables of a single parent solution. The UM operator mutates each index of a parent solution with some probability, and an index selected to mutate is assigned a value drawn uniformly at random within the upper and lower bounds of the decision variable.

Similarly, the polynomial mutation operator mutates each index of a single parent solution with some probability, but the indices selected to mutate are assigned values drawn from a distribution centered around the parent value.

We use the recommended parameters for each operator. For the SBX operator, the distribution index and crossover rate is set to 20 and 1.0, respectively, as used in NSGA-II [36]. The DE operator uses a scaling factor of 0.5 and a crossover rate of 1.0 similar to MOEA/D-DRA and MOEA/D-FRRMAB [109,194]. The UNDX, PCX, and SPX operators require three parent solutions, and $\sigma_\eta = 0.35/\sqrt{L}$ and $\sigma_\xi = 0.5$ for UNDX [99] where $L$ is the number of decision variables, $\sigma_\eta = \sigma_\zeta = 0.1$ for PCX as in experiments by Deb et al. [36], and the expanding rate is set to 1.0 for SPX uses as in experiments by Tsutsui et al. [176]. UM and polynomial mutation mutates each index with probability $1/L$, and polynomial mutation uses a distribution index of 20.

### 2.4.3 Comparative Study

A comparative study is conducted on 26 problems from the WFG [83], CEC 2009 [195], and DTLZ [41] test suites, and the performance of each AOS is recorded. The number of decisions variables $L$, the number of the objectives $m$, and the properties of each problem are summarized in Table 2.3. This diverse set of problems contains problems with different features that are considered difficult to solve including concavity, multimodality, discrete Pareto fronts, discontinuous Pareto front, and nonlinear objective functions.

*Experiment A*: The first experiment establishes a baseline performance and identifies the best operator for each MOEA on each problem. We hypothesize

Table 2.3: Test problem properties

| Problem | $L$ | $m$ | properties |
|---|---|---|---|
| WFG1 | 22 | 2 | mixed, separable, deceptive |
| WFG2 | 22 | 2 | nonseparable, discontinuous |
| WFG3 | 22 | 2 | nonseparable, degenerate |
| WFG4 | 22 | 2 | concave, multimodal, separable |
| WFG5 | 22 | 2 | concave, separable, deceptive |
| WFG6 | 22 | 2 | concave, nonseparable |
| WFG7 | 22 | 2 | concave, separable |
| WFG8 | 22 | 2 | concave, nonseparable |
| WFG9 | 22 | 2 | concave, multimodal, nonseparable |
| UF1 | 30 | 2 | complex Pareto set |
| UF2 | 30 | 2 | complex Pareto set |
| UF3 | 30 | 2 | complex Pareto set |
| UF4 | 30 | 2 | complex Pareto set, concave |
| UF5 | 30 | 2 | complex Pareto set, discrete |
| UF6 | 30 | 2 | complex Pareto set, discontinuous |
| UF7 | 30 | 2 | complex Pareto set |
| UF8 | 30 | 3 | complex Pareto set, concave |
| UF9 | 30 | 3 | complex Pareto set, discontinuous |
| UF10 | 30 | 3 | complex Pareto set, concave |
| DTLZ1 | 7 | 3 | multimodal, separable |
| DTLZ2 | 12 | 3 | concave, separable |
| DTLZ3 | 12 | 3 | concave, multimodal, separable |
| DTLZ4 | 12 | 3 | concave, biased |
| DTLZ5 | 12 | 3 | concave |
| DTLZ6 | 12 | 3 | concave, biased |
| DTLZ7 | 22 | 3 | discontinuous |

that the best operator will differ from problem to problem because an operator's performance is dependent on the problem at hand. Furthermore, on a given problem, we expect that an operator's behavior will vary with different MOEAs due to the ways each MOEA selects parent solutions for recombination and maintains the population. The best single-operator MOEA will be compared with the AOS

Table 2.4: Parameters for the MOEAs and operator selectors

| Parameter | Value |
|---|---|
| Neighborhood size $n_T$ | 20 |
| Probability of selecting mating pool $\delta$ | 0.9 |
| Crossover rate $CR$ | 1.0 |
| Maximum number of solutions an offspring can replace $n_R$ | 2 |
| Number of iterations until utility update | 50 |
| Minimum probability of selecting an operator $p_{min}$ | 0.1 |
| Adaptation rate $\alpha$ | 0.8 |
| Learning rate $\beta$ | 0.8 |
| IBEA scaling factor $\kappa$ | 0.5 |

in Experiment B to see if the combined use of a diverse set of operators leads to better or comparable performance than the best single operator.

*Experiment B*: The second experiment investigates the performance of each credit assignment paired with both PM and AP operator selectors. Each AOS is compared with 3 control strategies: 1) the default operator of MOEA/D-DRA, NSGA-II, and IBEA, which are DE, SBX, and SBX, respectively; 2) the best single-operator MOEA from Experiment A on each problem; and 3) an operator selector that randomly selects operators with uniform probability, to check whether intelligent operator selection is superior to random operator selection. We hypothesize that at least some of the AOS will be superior to the three control strategies.

Parameters for the MOEAs and operator selectors are shown in Table 2.4, which are the same as those from [202] and [109]. The population size and number of function evaluations for each problem is shown in Table 2.5, which is consistent with other comparative experiments [109, 111, 194]. Initial populations for each run are generated randomly.

*Performance Metrics*: Quality indicators are commonly used to measure the performance of MOEAs [28], and while a variety of indicators exists, no unary indicator can fully characterize an MOEAs performance [15]. Therefore, we use two popular indicators; the inverted generational distance (IGD) and hypervolume (HV).

The IGD indicator, shown in (2.21), computes the average of the minimum distance of each point in the true Pareto front, $PF^*$, to a point in $PF$, where the function $d(x, y)$ computes the Euclidean distance between the objective vectors $x$ and $y$. $PF^*$ for all problems were obtained through the solution database available within MOEAFramework. A lower IGD indicates that $PF$ is closer to $PF^*$.

$$IGD(PF, PF^*) = \frac{\sum_{x \in PF^*} (\min_{y \in PF} d(x, y))}{|PF^*|} \tag{2.21}$$

Unlike in IGD, $PF^*$ is not required to compute HV. The HV indicator simultaneously measures diversity and convergence by computing the closed hypervolume created in the objective space by a reference point, $z^*$, and the union of the all the solutions $x \in PF$. HV is given by (2.22), where $v(x, z^*)$ is the hypercube created with $x$ and $z^*$. Any point in $PF$ that is dominated by $z^*$ does not contribute to HV. Following the same approach presented in [111], the objective values of the solutions are normalized using the range of $PF^*$, and then the HV is computed using a reference point at $1.1^m$ as done in [111,112] and [86]. HV is computed using jMetal [46], which implements the fast hypervolume algorithm from the Walking Fish Group [182]. A large value for HV implies that $PF$ has good convergence and diversity.

$$HV(PF, z^*) = volume\Big( \bigcup_{x \in PF} v(x, z^*)\Big) \tag{2.22}$$

Table 2.5: Population size and number of function evaluations for each problem

| Problem | Population size | Function Evaluations |
|---|---|---|
| WFG | 100 | 25,000 |
| 2-objective UF | 600 | 300,000 |
| 3-objective UF | 1,000 | 300,000 |
| DTLZ | 105 | 30,000 |

To statistically compare the performance metrics of any two AOS, we conduct 30 independent runs for each problem on each experiment and use the Wilcoxon rank sum test with a significance level of 0.05. This non-parametric statistical test is commonly used to compare the performance of two algorithms [28] and can be applied to non-normal distributions. It ranks the measured performance of two AOS and compares the sum of ranks to see if the data from the two methods come from the same distribution (i.e. the performance of the two AOS are statistically equivalent).

## 2.5 Results

### 2.5.1 Experiment A

Our first experiment establishes a baseline performance of MOEA/D-DRA, IBEA, and NSGA-II with each of the six operators introduced earlier. Due to limited space, we only show the operators that achieved the best mean IGD and HV in Table 2.6 and Table 2.7, respectively.

The SBX operator performs well on many problems with all three MOEAs, especially on the WFG and DTLZ problems. Even when used with MOEA/D-DRA, SBX significantly outperforms the default DE operator on many of the WFG and DTLZ problems. The SBX operator is effective on separable problems [84], explaining its superior performance with two or more MOEAs in both metrics on the WFG1, WFG4, WFG7, DTLZ1, and DTLZ3 problems. Although DTLZ2 is also a separable problem, the SPX operator is competitive with SBX in the inverted generational distance metric. On the WFG6 and WFG8 problems, SBX is not the top performing operator since these problems are nonseparable, and instead, rotationally invariant operators such as DE and SPX are more effective. Surprisingly, SBX performs well on the other nonseparable problems, namely WFG2, WFG3 and WFG9. The SBX operator performs poorly on biased problems such as DTLZ4 and DTLZ6 that are designed to mislead an MOEA with nonuniform distribution of solutions in the objective space. On these problems, the DE operator, which can explore regions further from the parent solutions, outperforms the default IBEA and NSGA-II that use the SBX operator.

The problem properties can help explain the performance of the operators. Thus, if the properties of a problem are known in advance, they can help when selecting the MOEA and operator to solve the problem. Unfortunately, the problem properties are usually not well understood *a priori*, especially for real-world problems. Moreover, the problem properties do not always explain an operator's behavior as seen by the superior performance of SBX on the nonseparable WFG2, WFG3, and WFG9 problems.

Using the default operator for a given MOEA is not a robust strategy either because on many problems, it is outperformed by other operators in one or both metrics. Moreover, interactive effects between the operator and the MOEA significantly influence the search performance, but prior to the search, it is difficult to

Table 2.6: Experiment A: Best Operator and mean IGD

| Problem | MOEA/D-DRA [DE] | | NSGA-II [SBX] | | IBEA [SBX] | |
|---|---|---|---|---|---|---|
| WFG1 | PCX | 4.151E-1† | SBX | 4.195E-1 | SBX | 4.121E-1 |
| WFG2 | SBX | 1.056E-2† | SBX | 3.707E-3 | SBX | 1.846E-2 |
| WFG3 | SBX | 3.383E-2† | SBX | 3.442E-2 | SPX | 3.471E-2† |
| WFG4 | SBX | 5.961E-3† | SBX | 5.202E-3 | SBX | 1.023E-2 |
| WFG5 | DE | 2.773E-2 | SBX | 2.774E-2 | UNDX | 2.811E-2† |
| WFG6 | SPX | 2.007E-2 | DE | 7.203E-3† | DE | 1.279E-2† |
| WFG7 | SBX | 5.215E-3† | SBX | 5.788E-3 | UNDX | 5.520E-3† |
| WFG8 | DE | 3.487E-2 | DE | 3.205E-2† | DE | 3.606E-2† |
| WFG9 | SBX | 9.172E-3† | SBX | 8.843E-3 | UNDX | 1.028E-2† |
| UF1 | DE | 2.902E-3 | DE | 3.058E-2† | DE | 4.345E-2† |
| UF2 | DE | 1.126E-2 | UM | 1.788E-2† | UNDX | 2.758E-2† |
| UF3 | DE | 3.385E-2 | DE | 5.109E-2† | DE | 2.903E-2† |
| UF4 | UM | 3.587E-2† | UM | 3.057E-2† | UM | 3.586E-2† |
| UF5 | UM | 2.079E-1† | UM | 1.742E-1† | UM | 1.988E-1† |
| UF6 | SBX | 1.151E-1 | SBX | 1.024E-1 | UM | 1.441E-1 |
| UF7 | DE | 7.950E-3 | DE | 1.252E-2† | DE | 2.096E-2† |
| UF8 | DE | 5.502E-2 | UM | 1.050E-1† | SBX | 4.139E-1 |
| UF9 | SBX | 5.196E-2 | UM | 8.789E-2† | DE | 1.025E-1† |
| UF10 | SBX | 2.010E-1† | SBX | 2.436E-1 | UNDX | 4.111E-1† |
| DTLZ1 | SBX | 6.031E-2† | SBX | 9.289E-2 | SBX | 3.252E-1 |
| DTLZ2 | SBX | 6.739E-2† | SPX | 6.297E-2† | SPX | 8.200E-2† |
| DTLZ3 | SBX | 3.013E-1 | SBX | 4.840E0 | SBX | 6.469E-1 |
| DTLZ4 | DE | 5.464E-2 | UM | 4.157E-2† | DE | 5.032E-2† |
| DTLZ5 | DE | 1.547E-2 | SPX | 6.023E-3† | UNDX | 1.518E-2† |
| DTLZ6 | DE | 1.494E-2 | DE | 5.780E-3† | DE | 3.837E-2† |
| DTLZ7 | SBX | 1.192E-1 | SBX | 4.921E-2 | SBX | 9.992E-2 |

Using a Wilcoxon rank sum test with 0.05 significance level, † indicates significantly better IGD than the MOEA with its respective default operator. The default operator for each MOEA is given in the first row in [·].

assess if a given operator will help or hinder the MOEA. Fig. 2.2 shows boxplots of the hypervolume achieved by each single-operator MOEA on the WFG7 and DTLZ7 problems, where single-operator MOEAs are labeled as *MOEA-operator*. On the WFG7 problem, the UNDX operator performs the best with IBEA as opposed to the second and third worst with NSGA-II and MOEA/D-DRA, respec-

Table 2.7: Experiment A: Best Operator and mean HV

| Problem | MOEA/D-DRA [DE] | | NSGA-II [SBX] | | IBEA [SBX] | |
|---|---|---|---|---|---|---|
| WFG1 | SBX | 2.992E-1† | SBX | 3.133E-1 | SBX | 3.082E-1 |
| WFG2 | SBX | 7.725E-1† | SBX | 7.741E-1 | SBX | 7.706E-1 |
| WFG3 | SBX | 6.171E-1† | SBX | 6.164E-1 | SPX | 6.157E-1† |
| WFG4 | SBX | 4.310E-1† | SBX | 4.304E-1 | SBX | 4.246E-1 |
| WFG5 | PCX | 4.006E-1† | SBX | 4.042E-1 | PCX | 4.025E-1† |
| WFG6 | SPX | 3.886E-1 | DE | 4.129E-1† | DE | 4.072E-1† |
| WFG7 | SBX | 4.190E-1† | SBX | 4.173E-1 | UNDX | 4.154E-1† |
| WFG8 | DE | 3.497E-1 | DE | 3.513E-1† | DE | 3.451E-1† |
| WFG9 | SBX | 4.510E-1† | SBX | 4.560E-1 | PCX | 4.507E-1 |
| UF1 | DE | 8.716E-1 | DE | 8.246E-1† | DE | 8.145E-1† |
| UF2 | DE | 8.592E-1 | UM | 8.504E-1† | UNDX | 8.379E-1 |
| UF3 | DE | 8.287E-1 | DE | 7.846E-1† | DE | 8.262E-1† |
| UF4 | UM | 4.746E-1† | UM | 4.867E-1† | UM | 4.807E-1† |
| UF5 | UM | 3.598E-1 | UM | 3.966E-1† | UM | 3.685E-1† |
| UF6 | SBX | 4.306E-1 | SBX | 4.485E-1 | SBX | 4.226E-1 |
| UF7 | DE | 6.944E-1 | DE | 6.873E-1† | DE | 6.787E-1† |
| UF8 | DE | 6.881E-1 | UM | 5.721E-1† | UNDX | 4.638E-1 |
| UF9 | DE | 1.007E0 | UM | 9.207E-1† | DE | 9.389E-1† |
| UF10 | SBX | 3.823E-1† | UNDX | 3.352E-1† | UNDX | 3.059E-1† |
| DTLZ1 | SBX | 1.084E0† | SBX | 1.027E0 | SBX | 6.073E-1 |
| DTLZ2 | SBX | 7.127E-1† | SBX | 7.037E-1 | SBX | 7.415E-1 |
| DTLZ3 | SBX | 5.104E-1† | DE | 9.803E-3† | SBX | 1.680E-1 |
| DTLZ4 | DE | 6.862E-1 | SBX | 7.096E-1 | DE | 7.390E-1† |
| DTLZ5 | SBX | 2.621E-1† | SBX | 2.667E-1 | SPX | 2.638E-1† |
| DTLZ6 | DE | 2.643E-1 | DE | 2.696E-1† | DE | 2.604E-1† |
| DTLZ7 | SBX | 4.661E-1† | SBX | 5.383E-1 | SBX | 5.222E-1 |

Using a Wilcoxon rank sum test with 0.05 significance level, † indicates significantly better IGD than the MOEA with its respective default operator. The default operator for each MOEA is given in the first row in [·].

tively. On the DTLZ7 problem, the same UNDX operator achieves the second best median hypervolume with NSGA-II but the worst with MOEA/D-DRA and third worst with IBEA.

The results from Experiment A illustrate the unintuitive, yet significant, consequences of selecting an effective combination of an operator and MOEA for a given

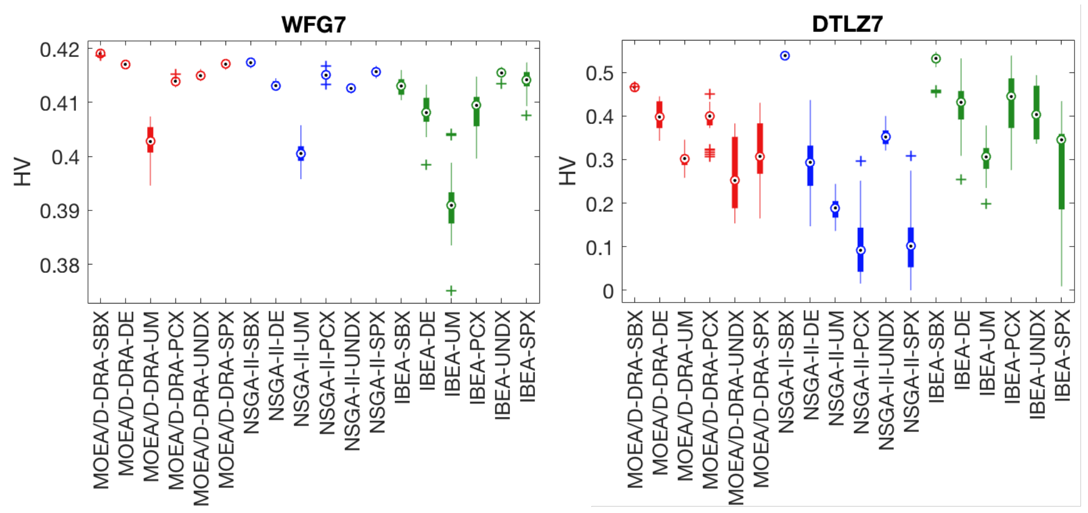

Figure 2.2: Boxplot of HV achieved by single operator MOEAs on the WFG7 and DTLZ7 problems.

problem. A single-operator MOEA that works well on one problem does not work well on another, supporting the No Free Lunch Theorem of Optimization [187], which states that a heuristic optimization method works well for only a subset of problem instances. This provides the motivation for AOS, which attempts to elevate the generality of an MOEA in order to be effective across more problems.

### 2.5.2 Experiment B

#### Decomposition-based Credit Assignment

First, we examine the performance of the decomposition-based credit assignments. Table A.1 and Table A.2 in Appendix A show the performance metrics of each decomposition-based AOS compared to the default operator MOEA/D-DRA (i.e. DE). Table 2.8 and Table 2.9 summarize how the IGD and HV of each decomposition-based AOS statistically compares to the default operator, the best

Table 2.8: Statistical comparison of the final IGD achieved by decomposition-based AOS and the control strategies

| | Default Operator | | | Best Operator | | | Random | | |
|---|---|---|---|---|---|---|---|---|---|
| | + | = | - | + | = | - | + | = | - |
| Random | 11 | 6 | 9 | 2 | 10 | 14 | | | |
| PM-OP-De | 11 | 6 | 9 | 4 | 7 | 15 | 2 | 24 | 0 |
| PM-SI-De | 12 | 6 | 8 | 3 | 11 | 12 | 8 | 18 | 0 |
| PM-CS-De | 11 | 5 | 10 | 5 | 6 | 15 | 7 | 16 | 3 |
| AP-OP-De | 10 | 5 | 11 | 4 | 7 | 15 | 1 | 23 | 2 |
| AP-SI-De | 11 | 7 | 8 | 3 | 12 | 11 | 7 | 18 | 1 |
| AP-CS-De | 11 | 5 | 10 | 4 | 8 | 14 | 6 | 19 | 1 |

Using a Wilcoxon rank sum test with 0.05 significance level, +, =, - indicate the number of problems that the corresponding AOS performed statistically better, equivalent, or worse than the control strategy.

single-operator from Experiment A, and the random operator selector. Each AOS is denoted as *operator selector-credit assignment* such as PM-OP-De for probability matching with the decomposition-based OP-De credit assignment.

Table 2.8 and Table 2.9 show that the performance of a decomposition-based AOS with PM is similar to that with AP. Additionally, all of the decomposition-based AOS are competitive with the default operator MOEA/D-DRA in the inverted generational distance metric, but superior in the hypervolume metric. Experiment A showed that the best operator for MOEA/D-DRA changed from problem to problem, but the decomposition-based AOS are able to leverage the diversity in the operators to outperform the default MOEA/D-DRA.

The decomposition-based AOS are outperformed by the best single-operator MOEA/D-DRA, but in general, they achieve better performance than the majority of the single-operator MOEA/D-DRA strategies. For example, Fig. 2.3 shows the hypervolume achieved on the WFG7 and DTLZ7 problems by each single-operator MOE/D-DRA, the random operator selector, and each decomposition-

Table 2.9: Statistical comparison of the final HV achieved by decomposition-based AOS and the control strategies

| | Default Operator | | | Best Operator | | | Random | | |
|---|---|---|---|---|---|---|---|---|---|
| | + | = | - | + | = | - | + | = | - |
| Random | 14 | 4 | 8 | 3 | 7 | 16 | | | |
| PM-OP-De | 14 | 4 | 8 | 5 | 4 | 17 | 1 | 25 | 0 |
| PM-SI-De | 16 | 4 | 6 | 4 | 10 | 12 | 11 | 15 | 0 |
| PM-CS-De | 14 | 4 | 8 | 5 | 7 | 14 | 7 | 17 | 2 |
| AP-OP-De | 15 | 2 | 9 | 5 | 4 | 17 | 2 | 21 | 3 |
| AP-SI-De | 15 | 5 | 6 | 5 | 9 | 12 | 11 | 14 | 1 |
| AP-CS-De | 14 | 4 | 8 | 5 | 8 | 13 | 7 | 18 | 1 |

Using a Wilcoxon rank sum test with 0.05 significance level, +, =, - indicate the number of problems that the corresponding AOS performed statistically better, equivalent, or worse than the control strategy.

based AOS. It shows the decomposition-based AOS outperforming all single-operator MOEA/D-DRA except the one with SBX, the best operator for these problems. Moreover, the decomposition-based AOS outperforms the best single-operator MOEA/D-DRA in one or both metrics on a few problems, namely WFG1, UF5, DTLZ4, and DTLZ7. This is significant because the AOS does not use any *a priori* knowledge of the best operator for MOEA/D-DRA, which is different from problem to problem as seen in Experiment A. Furthermore, the statistically significant improvement in the search performance over the single-best operator implies that there are positive interactions between some operators.

The relatively high performance of the random operator selector supports the hypothesis that positive interactions exist between the operators. While the AOS with SI-De and CS-De credit assignments perform better than the random operator selector overall, it is interesting to see the random operator selector perform comparably to the decomposition-based AOS on many problems as seen in Table 2.8 and Table 2.9.

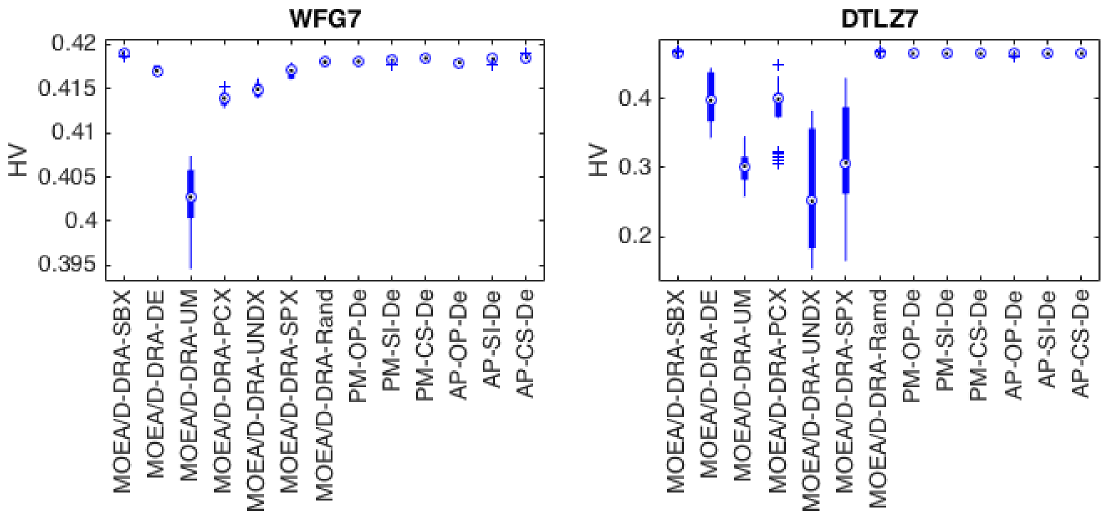

Figure 2.3: Boxplot of HV achieved on the WFG7 and DTLZ7 problems by single-operator MOEA/D-DRA, random operator selector with MOEA/D-DRA (MOEA/D-DRA-Rand), and decomposition-based AOS.

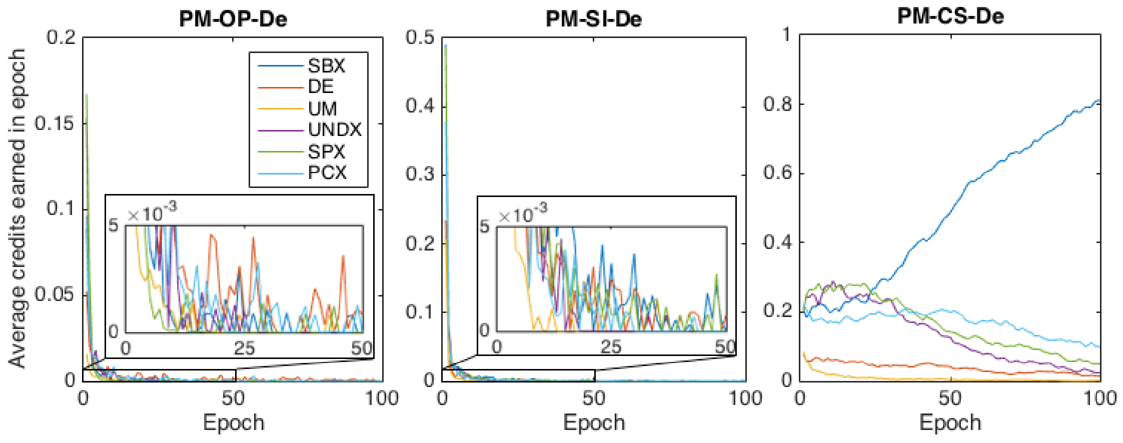

Figure 2.4: The average credits received per epoch by each operator during the search on the WFG7 problem using dominance-based credits and PM. An epoch is equivalent to 250 evaluations. Note the difference in scale for each AOS.

Of the three decomposition-based credit assignments, OP-De is the most statistically similar to the random operator selector. One explanation for the similarity is that OP-De does not provide the operator selector with sufficient information about an operator's impact because OP-De rewards operators based only on local improvements. Fig. 2.4 shows the average credits received by each operator every epoch, or 250 evaluations, averaged over 30 runs on the WFG7 problem when using PM-OP-De. It shows that PM-OP-De rewards the UM and SPX operators the least even though the SPX operator performs well on this problem as shown in Fig. 2.3. For the rest of the search, all the other operators are rewarded similarly, resulting in statistically similar performance to the random operator selector. We can conclude from Experiment A, that the SBX, DE, and SPX operators perform well with MOEA/D-DRA on the WFG7 problem because they can replace incumbent solutions in neighboring subproblems as well as the current subproblem, but since the OP-De credit assignment only rewards operators for their local impact, it does not take into account this effect on neighboring subproblems.

Contrary to the OP-De credit assignment, SI-De rewards operators for creating offspring that replace the incumbent and neighboring solutions to a subproblem, and it performs better overall than the random operator selector in the hypervolume metric. This broader perspective helps the AOS with SI-De to outperform the random operator selector on 11 out of the 26 problems when used with the PM or AP operator selector. An inspection of the credit history on the WFG7 problem, shown in Fig. 2.4, reveals that improvements to the neighborhood are only incremental after the 10th epoch. The AOS is able to discriminate UM as the worst operator and it receives few to zero credits during the search. After the 20th epoch, however, it is difficult to see any differentiation between the credits rewarded to the remaining operators. As suggested by the random operator selector, a good strategy on this problem is to use a uniform combination of the operators, and the SI-De credit assignment signals the operator selector to begin using a uniform combination of operators. The AOS with SI-De significantly outperforms the random operator selector in both metrics most likely because SI-De quickly identifies UM's poor performance.

CS-De, which rewards operators for creating many solutions in the neighborhood of a subproblem, also outperforms the random operator selector on several problems. It does not perform as well as the AOS with SI-De, especially in the hypervolume metric, most likely because older solutions created in previous iterations are viewed as equally important as newly created ones when using CS-De. This introduces a lag in the credit assignment's ability to track an operator's performance. For example, Fig. 2.4 shows that the SBX, UNDX, SPX, and PCX operators are rewarded similarly for the first third of the search, and it takes some time before SBX, the top performing operator on this problem, begins to receive the most credits. The slow dynamics is due to the implemented version of MOEA/D-DRA that only allows up to 2 solutions to be replaced, so the other solutions created by the UNDX, SPX, and PCX operators remain in the neighborhood and continue to contribute towards the rewarded credits. Moreover, solutions created by UM, the worst performing operator on this problem, remain in the neighborhoods, so UM continues to receive credits for a significant portion of the search. Note that as SBX replaces solutions in the neighborhood the credits of the other operators necessarily decrease because each solution in the fixed-sized neighborhood corresponds to credit rewarded to only one operator.

### Dominance-based Credit Assignment

Table A.3 and Table A.4 in Appendix A show the performance metrics of each dominance-based AOS and their performance compared to the default operator NSGA-II (i.e. SBX). Table 2.10 and Table 2.11 summarize how the IGD and HV of each dominance-based AOS statistically compares to the default operator, the best single-operator from Experiment A, and the random operator selector.

The dominance-based AOS generally perform worse than the best single-operator NSGA-II but are competitive with the default NSGA-II. Experiment A showed that the SBX operator with NSGA-II achieved the best performance on roughly half of the problems, so the dominance-based AOS have a more difficult time outperforming the default NSGA-II. The poor performance of the dominance-based AOS relative to the best single-operator NSGA-II is partially due to large differences in performance between the best and worst operators. For example, on the WFG7 or DTLZ7 problems, NSGA-II with the SBX operator achieves a significantly larger hypervolume than with any of the other operators. The AOS must learn to use SBX more often than the others, but in the learning process, the AOS must use all operators multiple times and expend evaluation functions. Consequently, the AOS do not perform as well as the best operator on the WFG7 or DTLZ7 problems, but as with the decomposition-based AOS, the application of AOS on NSGA-II elevates the generality of the algorithm as shown in Fig. 2.5. Despite the presence of poorly performing operators, each AOS achieves a similar or better hypervolume to the SBX operator compared to the second best single-operator NSGA-II.

While the random operator selector performs relatively well on all problems, it is significantly outperformed by each dominance-based AOS. The random operator selector is even outperformed by AOS with OP-Do. OP-Do uses very local information to reward an operator for creating offspring that dominates their parents, so we had expected it to perform poorly like the OP-De credit assignment. Instead, AOS with OP-Do are the top performing dominance-based AOS most likely

Table 2.10: Statistical comparison of the final IGD achieved by dominance-based AOS and the control strategies

| | Default Operator | | | Best Operator | | | Random | | |
|---|---|---|---|---|---|---|---|---|---|
| | + | = | - | + | = | - | + | = | - |
| Random | 9 | 4 | 13 | 2 | 5 | 19 | | | |
| PM-OP-Do | 11 | 6 | 9 | 3 | 7 | 16 | 11 | 13 | 2 |
| PM-SI-Do | 10 | 6 | 10 | 3 | 8 | 15 | 8 | 14 | 4 |
| PM-CS-Do | 10 | 5 | 11 | 3 | 8 | 15 | 13 | 11 | 2 |
| AP-OP-Do | 12 | 5 | 9 | 5 | 6 | 15 | 13 | 11 | 2 |
| AP-SI-Do | 10 | 5 | 11 | 2 | 8 | 16 | 10 | 15 | 1 |
| AP-CS-Do | 11 | 6 | 9 | 4 | 7 | 15 | 13 | 10 | 3 |

Using a Wilcoxon rank sum test with 0.05 significance level, +, =, - indicate the number of problems that the corresponding AOS performed statistically better, equivalent, or worse than the control strategy.

Table 2.11: Statistical comparison of the final HV achieved by dominance-based AOS and the control strategies

| | Default Operator | | | Best Operator | | | Random | | |
|---|---|---|---|---|---|---|---|---|---|
| | + | = | - | + | = | - | + | = | - |
| Random | 9 | 3 | 14 | 3 | 2 | 21 | | | |
| PM-OP-Do | 9 | 8 | 9 | 2 | 9 | 15 | 14 | 10 | 2 |
| PM-SI-Do | 9 | 4 | 13 | 2 | 8 | 16 | 9 | 14 | 3 |
| PM-CS-Do | 8 | 7 | 11 | 1 | 10 | 15 | 16 | 7 | 3 |
| AP-OP-Do | 9 | 9 | 8 | 2 | 10 | 14 | 18 | 7 | 1 |
| AP-SI-Do | 9 | 5 | 12 | 3 | 8 | 15 | 10 | 14 | 2 |
| AP-CS-Do | 10 | 3 | 13 | 4 | 6 | 16 | 12 | 12 | 2 |

Using a Wilcoxon rank sum test with 0.05 significance level, +, =, - indicate the number of problems that the corresponding AOS performed statistically better, equivalent, or worse than the control strategy.

because OP-Do prefers operators that push convergence over those that increase diversity and the diversity management is left to underlying MOEA. Fig. 2.6 offers a closer look at the credits rewarded by the dominance-based credit assignments on the DTLZ7 problem, where each epoch is 300 evaluations. OP-Do correctly identifies SBX as the best operator and rewards it the most throughout the entire search, but a small amount of credit is rewarded to the other operators, signaling the operator selector to continually explore the other operators. This explains why AOS with the OP-Do credit assignment outperforms the random operator selector, but cannot outperform NSGA-II with SBX on this problem.

AOS strategies using the SI-Do credit assignment do not perform as well as AOS with OP-Do or CS-Do credit assignments, but it still outperforms the random operator selector. One explanation for the lower performance with the SI-Do credit assignments is that it rewards an operator for creating a solution that enters the Pareto front but does not consider if the new solution adds to the diversity of the front. A solution that enters the Pareto front that crowds an existing solution is less desirable than one that enters the Pareto front in a sparsely populated region, but SI-Do rewards both solutions equally. Fig. 2.6 shows that the SI-Do credit assignment identifies SBX as one of the top performing operators but not the best. Instead, SI-Do rewards the PCX operator the most, despite its poor performance on this problem as shown in Fig. 2.5. We can infer from the credit history of PM-OP-Do that the PCX operator does not consistently create offspring that dominate its parents. In addition, PCX creates offspring near the parent solutions. Therefore, the lower performance of the AOS with SI-Do on the DTLZ7 problem is likely due to PCX inserting solutions in an already occupied region of the Pareto front.

The AOS with the CS-Do credit assignment, which rewards an operator for its contribution of solutions to the Pareto front, perform almost as well as those with OP-Do. CS-Do is more robust to being misguided by crowded solutions on the Pareto front even though it is similar to SI-Do and does not estimate density metrics. In Fig. 2.6, CS-Do initially identifies SBX as the best performing operator, but solutions created by the PCX operator gradually increase their presence in the

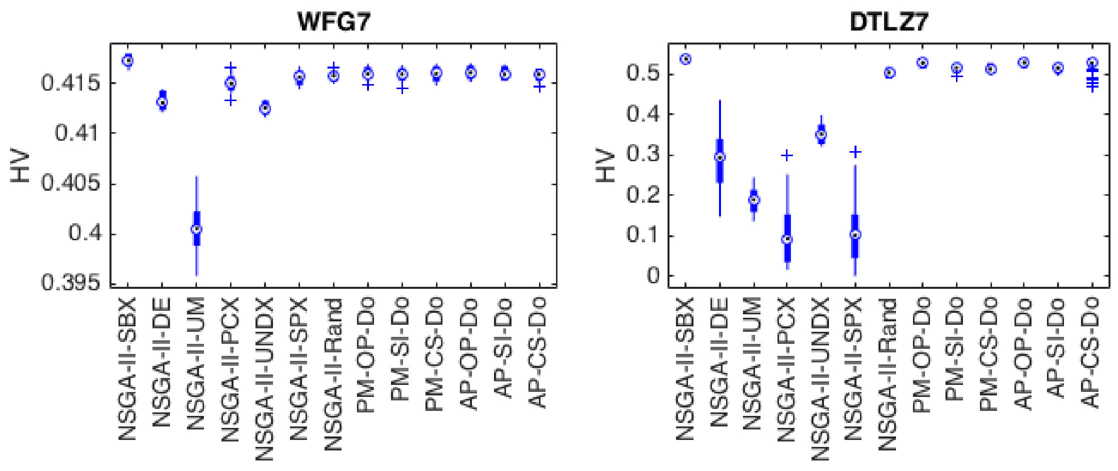

Figure 2.5: Boxplot of HV achieved on the WFG7 and DTLZ7 problems by the single-operator NSGA-II, the random operator selector with NSGA-II (NSGA-II-Rand), and the dominance-based AOS.

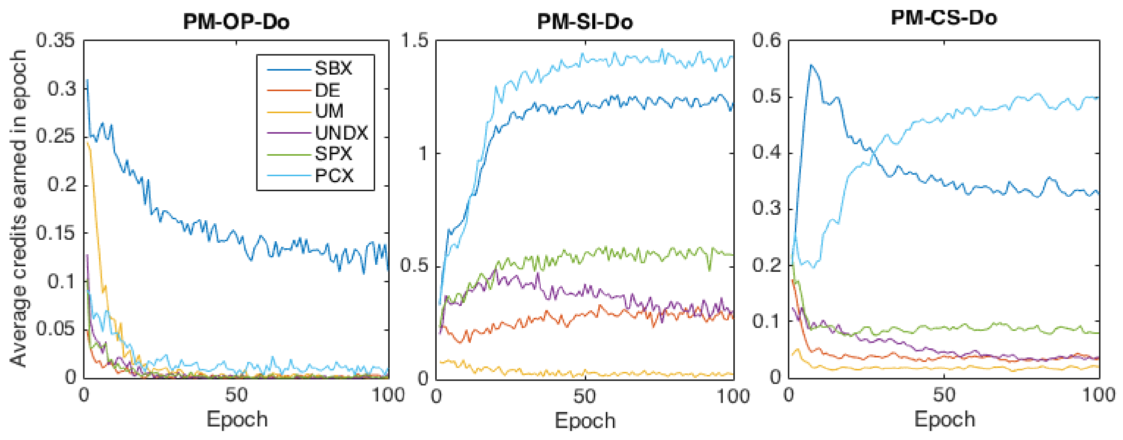

Figure 2.6: The average credits received per epoch by each operator during the search on the DTLZ7 problem using dominance-based credits and PM. An epoch is equivalent to 300 evaluations. Note the difference in scale for each AOS.

Pareto front until about half of the Pareto front consists of solutions created by PCX. If the PCX operator is indeed inserting solutions in crowded regions of the Pareto front, after some time, it will begin replacing its own solutions preventing PCX from earning a disproportionate amount of credits. From the credit history of PM-SI-Do, we see that the PCX and SBX operators consistently insert solutions into the Pareto front, but the credit history of PM-CS-Do exhibits stability in the credits rewarded to all operators in the second half of the search. Therefore, the SBX and PCX operators must be replacing their own solutions on the Pareto front and not significantly changing each operator's contribution to the Pareto front.

### Indicator-based Credit Assignment

Table A.5 and Table A.6 in Appendix A show the performance metrics of each indicator-based AOS and their performance compared to the default operator IBEA (i.e. SBX). Table 2.12 and Table 2.13 summarize how the IGD and HV of each indicator-based AOS statistically compares to the default operator, the best single-operator from Experiment A, and the random operator selector.

Table 2.12 and Table 2.13 shows that the indicator-based AOS perform significantly better than the default-operator IBEA in both performance metrics. However, as with decomposition-based and dominance-based AOS, the indicator-based AOS are significantly outperformed by the best single-operator IBEA on the majority of the problems. Nonetheless, the indicator-based AOS elevate the performance of the MOEA toward that of the single-best operator, which is also consistent with the effect of the decomposition-based and dominance-based AOS. Fig. 2.7 shows the achieved hypervolumes on the WFG7 and DTLZ7 problems of the single-operator IBEA, IBEA with the random operator selector, and the indicator-based AOS, and it shows the indicator-based AOS performing well despite the inclusion of poor performing operators in the operator set such as UM on the WFG7 problem or UM and SPX on the DTLZ7 problem. Interestingly, Table 2.12, Table 2.13, and Fig. 2.7 show that, once again, the random operator selector

Table 2.12: Statistical comparison of the final IGD achieved by indicator-based AOS and the control strategies

| | Default Operator | | | Best Operator | | | Random | | |
|---|---|---|---|---|---|---|---|---|---|
| | + | = | - | + | = | - | + | = | - |
| Random | 13 | 5 | 8 | 3 | 3 | 20 | | | |
| PM-OP-I | 12 | 7 | 7 | 3 | 3 | 20 | 4 | 20 | 2 |
| PM-SI-I | 12 | 6 | 8 | 2 | 4 | 20 | 3 | 23 | 0 |
| PM-CS-I | 13 | 9 | 4 | 3 | 5 | 19 | 8 | 18 | 0 |
| AP-OP-I | 12 | 9 | 5 | 2 | 6 | 18 | 4 | 20 | 2 |
| AP-SI-I | 12 | 7 | 7 | 3 | 2 | 21 | 5 | 21 | 0 |
| AP-CS-I | 13 | 7 | 6 | 3 | 5 | 18 | 7 | 19 | 0 |

Using a Wilcoxon rank sum test with 0.05 significance level, +, =, - indicate the number of problems that the corresponding AOS performed statistically better, equivalent, or worse than the control strategy.

Table 2.13: Statistical comparison of the final HV achieved by indicator-based AOS and the control strategies

| | Default Operator | | | Best Operator | | | Random | | |
|---|---|---|---|---|---|---|---|---|---|
| | + | = | - | + | = | - | + | = | - |
| Random | 13 | 8 | 5 | 3 | 7 | 16 | | | |
| PM-OP-I | 15 | 5 | 6 | 3 | 9 | 14 | 7 | 17 | 2 |
| PM-SI-I | 16 | 6 | 4 | 4 | 7 | 15 | 4 | 22 | 0 |
| PM-CS-I | 18 | 5 | 3 | 4 | 8 | 14 | 9 | 17 | 0 |
| AP-OP-I | 15 | 5 | 6 | 3 | 7 | 16 | 7 | 16 | 3 |
| AP-SI-I | 16 | 5 | 5 | 4 | 6 | 16 | 6 | 20 | 0 |
| AP-CS-I | 17 | 5 | 4 | 4 | 7 | 15 | 13 | 13 | 0 |

Using a Wilcoxon rank sum test with 0.05 significance level, +, =, - indicate the number of problems that the corresponding AOS performed statistically better, equivalent, or worse than the control strategy.

performs well relative to the other indicator-based AOS, further confirming that a diverse set of operators improves the robustness of the algorithm.

Similar to the AOS with OP-Do, AOS using OP-I outperforms the random

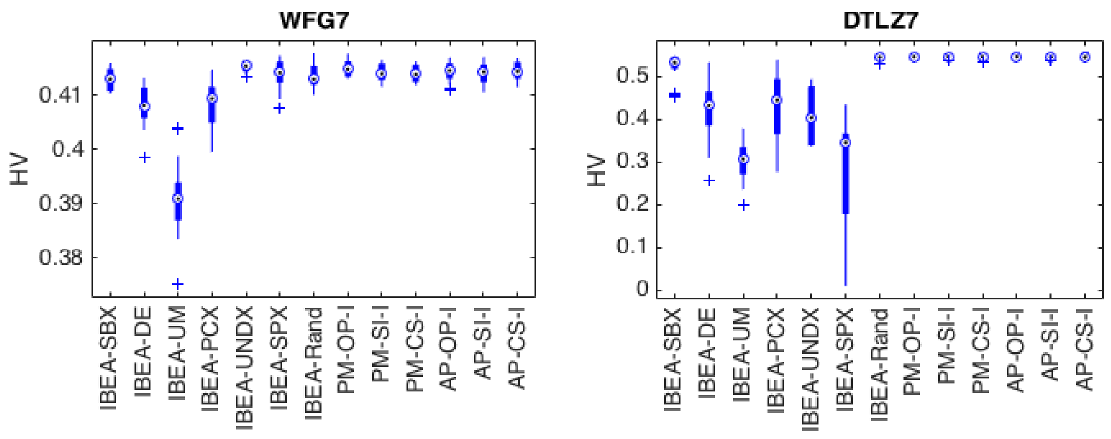

Figure 2.7: Boxplot of HV achieved on the WFG7 and DTLZ7 problems by single-operator IBEA, random operator selector with IBEA (IBEA-Rand), and indicator-based AOS.

operator selector on a handful of problems despite rewarding an operator based on its local impact. On several problems, however, OP-I incorrectly categorizes the top performing operators as poor performers, which causes PM-OP-I and AP-OP-I to be significantly outperformed by the random operator selector on the WFG1 and UF4 problems. Fig. 2.8 shows the credit history rewarded by OP-I on the DTLZ7 problem using the PM operator selector. OP-I identifies SPX as the best performing operator although it is one of the worst performers on this problem as shown in Fig. 2.7. In addition, OP-I views SBX, the best performing operator on this problem, as mediocre. Despite being misguided, AOS with OP-I still has statistically equivalent performance as the random operator selector on DTLZ7, which shows that using a diverse set of operators adds robustness to poor selection of the operators.

The SI-I credit assignment has better accuracy in identifying the top performing operators, which prevents AOS with SI-I from performing worse than the random operator selector. In Fig. 2.8, we see that PM-SI-I on the DTLZ7 problem correctly

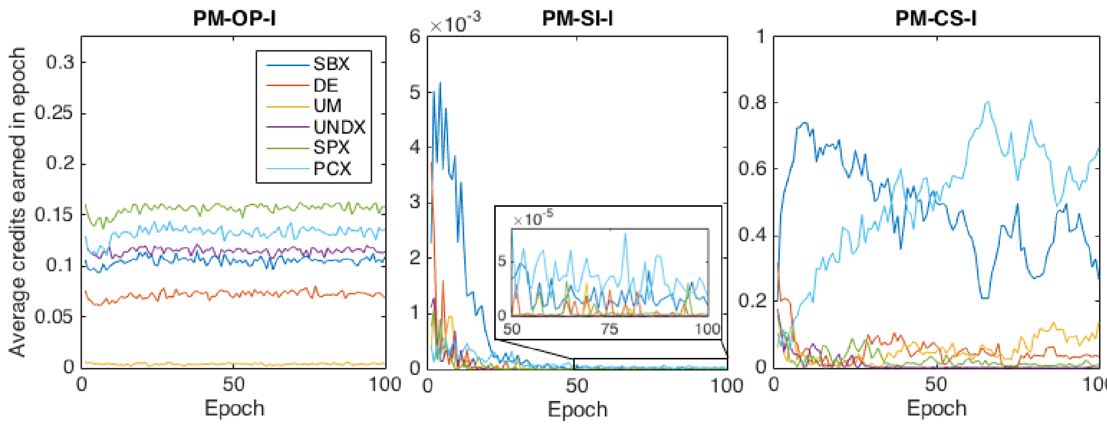

Figure 2.8: The average credits received per epoch by each operator during the search on the DTLZ7 problem using indicator-based credits and PM. An epoch is equivalent to 300 evaluations. Note the difference in scale for each AOS.

identifies SBX as the best operator at the beginning of the search. However, as in the case of the dominance-based SI-Do credit assignment, the PCX operator eventually surpasses SBX in received credits. Again, the PCX operator exploits the high quality solutions created by the SBX operator and is able to more frequently create new offspring that contribute to the R2 indicator value. Thus, the SBX operator is selected less frequently than optimal, but the early detection of high performing operators allows AOS with SI-I to perform somewhat better than the random operator selector in general.

Examination of the credit history of PM-CS-I in Fig. 2.8 confirms that the PCX operator on the DTLZ7 problem takes advantage of the high quality solutions produced by the SBX operator. After the 40th epoch, the credit history of the PCX operators mirrors that of SBX, indicating that the PCX operator is replacing solutions created by SBX in the set that contributes to the R2 indicator. Unlike in PM-SI-I, with CS-I, there is a clearer distinction between the credits rewarded to the SBX or PCX operators and the credits rewarded to the other lower performing operators, which helps the AOS with CS-I to outperform the random operator selector to a greater degree. In fact, the AOS using CS-I perform the best out of all indicator-based AOS.

## 2.6 Discussion

Of the decomposition-based AOS presented in this paper SI-De and CS-De outperformed the default operator MOEA/D-DRA in the hypervolume metric and the random operator selector in both performance metrics. The OP-De credit assignment, which rewards operators that improve the solution of the current subproblem, led to statistically similar performance to the random operator selector because it relies on local information. SI-De and CS-De leverage a broader perspective of an operator's impact and reward operators that improve the solutions of the neighboring subproblem and that contribute solutions to the neighborhood, respectively. The performance of SI-De is the best amongst the presented decomposition-based credit assignments. Moreover, the values used by SI-De are already computed within MOEA/D-DRA so it requires little computational overhead. A potential improvement upon CS-De would be to increase the number of solutions an offspring can replace to get a broader measure of an operator's impact, which could speed up the response of the credit assignment to the dynamic state of the evolving population. Unfortunately, the search performance of MOEA/D-DRA is sensitive to the neighborhood size [85,196], so while a larger neighborhood may benefit the CS-De credit assignment, it may be detrimental to the general performance of the AOS.

All dominance-based credit assignments are competitive with the default operator NSGA-II but superior to the random operator selector in general. Contrary to the decomposition-based credit assignments, the dominance-based OP-Do, which rewards operators for creating offspring that dominate their parent, uses the most local information but leads to better performance than SI-Do. Since OP-Do performs well on the tested problems and has the least computational complexity of the presented dominance-based credit assignments, we recommend using OP-Do credit assignments for problems with two or three objectives. The performance of AOS using SI-Do, which rewards operators for inserting offspring into the Pareto front, or CS-Do, which rewards operators for their contribution of solutions to the Pareto front, are hindered by operators that insert solutions into crowded areas of the Pareto front. This can be prevented by incorporating a density metric such as crowding distance or using a modified dominance-relation such as $\epsilon$-dominance [105] in these credit assignments. Furthermore, the addition of a density metric or modified dominance-relation may be necessary in many-objective problems to counter the loss in selection pressure of Pareto-dominance relationships in a high-dimensional objective space [110, 112, 179].

All indicator-based credit assignments outperform both the default operator IBEA and the random operator selector in both metrics. OP-I, which rewards operators for creating offspring with a high binary indicator value when compared to their parent, is the only indicator-based credit assignment to be outperformed by the random operator selector on a few problems because it relies on highly localized information and inaccurately identifies the top performing operators. SI-I and CS-I, which reward operators for creating offspring that improve and contribute to the R2 indicator value, respectively, suffer from similar issues as SI-Do and CS-Do, where the credit assignments reward operators for inserting offspring into crowded regions of the objective space. Since SI-I and CS-I both have the same computational complexity and AOS with CS-I perform better, CS-I is recommended over SI-I. Indicator-based credit assignments are expensive to compute, however, so we recommend using indicators with low computational complexity or a solution set that contains few individuals.

In summary, we offer a few general statements about AOS using multiobjective credit assignments.

- **An AOS elevates the generality of an algorithm.** The goal of an AOS is to raise the generality of an algorithm so that it can perform well over a greater number of problems. The results in this paper show that the AOS consistently outperform most of the single-operator MOEAs. AOS also significantly improves upon the performance of the default versions of MOEA/D-DRA and IBEA.
- **An AOS can outperform the best single-operator MOEA.** While the AOS were outperformed by the best single-operator MOEA in general, all AOS were able to outperform the best single-operator MOEA on a few problems. This indicates that some operators are complementary and search cooperatively.
- **Outperforming random operator selection is nontrivial.** The results show that simply using a diverse set of operators randomly, without an intelligent operator selector, can also lead to improved performance. Past studies on single-objective AOS have also observed the random operator selector performing well [70, 134, 145]. The results show the random operator selector performing well in many problems compared to the tested AOS and even outperformed the best single-operator MOEA on a few problems. The diverse set of operators increases the exploration and exploitation capabilities

of the MOEA because solutions can be translated using multiple methods through the decision space. The robustness in the search performance despite selecting poor operators is likely due to the MOEA's effective selection and retention of the high quality solutions.

- **Multiobjective AOS can outperform random operator selection.** Of the 9 credit assignments presented in this paper, all but one (i.e. OP-De) were able to significantly outperform the random operator selector. The other 8 credit assignments, namely SI-De, CS-De, OP-Do, SI-Do, CS-Do, OP-I, SI-I and CS-I, should be examined further in future work.

## 2.7 Conclusion

This paper, for the first time, introduced a classification of multiobjective credit assignments for AOS in order to identify the main components of multiobjective credit assignments and study their effects on the efficacy of measuring an operator's impact when solving a multiobjective problem. The classification is based on 1) the solutions or solution sets used to assess an operator's impact (e.g. parent solution, Pareto front, neighborhood) and 2) the MOEA fitness function used to compare the quality of solutions. 9 credit assignment classes were identified with research gaps in 5 of the 9, and to fill in the gaps, we proposed new decomposition-based credit assignments (i.e. OP-De, CS-De) and indicator-based credit assignments (i.e. OP-I, SI-I, and CS-I). A total of 9 multiobjective credit assignments spanning all categories were examined through a comparative study.

We showed that a diverse set of operators with a random operator selector can consistently perform better than many of the single-operator MOEAs and that it is nontrivial to outperform this random operator selector on standard benchmarking problems. The credit assignments that succeeded in outperforming the random operator selector should be investigated further. These include SI-De, CS-De, OP-Do, SI-Do, CS-Do, OP-I, SI-I and CS-I credit assignments.

The operators used in this paper were selected based on their popularity and diversity, but they may have overlapping behaviors which can prevent the operator selector from distinguishing one operator from another. Redundant operator behaviors on a problem may have contributed to the similar performance of the random operator selector and some of the AOS. The credit assignments in this paper should be applied to different sets of operators to generalize the findings.

Finally, AOS with credit assignments introduced in this paper should be applied on real-world problems because standard benchmarking problems are not necessarily representative of real-world problems. Real-world problems may reveal more difficulties or opportunities to modify and improve the credit assignments.

CHAPTER 3

# INCORPORATING EXPERT KNOWLEDGE INTO EVOLUTIONARY ALGORITHMS WITH OPERATORS AND CONSTRAINTS TO DESIGN SATELLITE SYSTEMS

## 3.1 Introduction

Evolutionary algorithms (EA) are popular optimization algorithms to solve real-world design problems because they can be applied to problems with non-linear, non-convex, and non-differentiable properties in addition to problems with multiple incommensurable and conflicting objectives [28]. However, EAs can be considered computationally inefficient since they rely on many function evaluations to discover high-quality solutions for the problem at hand, and if each evaluation requires computationally expensive or time-consuming simulations, evaluating thousands of solutions is impractical [22, 118].

A well-known approach to improve the search performance of an EA on a given problem is to incorporate relevant domain-specific or expert knowledge into the EA [33, 66, 74, 129, 152, 187]. According to the No Free Lunch Theorem (NFLT) for optimization [187], an EA that does not make any assumptions or use knowledge about a given problem has the same expected search performance as any other black-box search algorithm, including a random search [80]. On the other hand, by taking advantage of available expert knowledge and using it to guide the optimization, a knowledge-intensive EA is able to outperform a knowledge-independent one and find high-quality solutions with fewer function evaluations [13, 22, 77, 118].

If EAs are to tackle complex problems that have computationally expensive evaluation functions, it will be beneficial for the EA to leverage available knowledge from domain-experts. There are three main issues that need to be addressed, however, when incorporating knowledge into EAs. First, it has been shown that not all available knowledge is equally beneficial in improving the search performance [22, 77]. The value that domain-specific knowledge adds to the search can depend not only on its quality, but also on the state of the search and the ability of the knowledge to produce improving solutions [75, 77]. Second, design heuristics from experts typically focus on improving just one of the multiple objectives, and obtaining knowledge from multiple experts or other sources can introduce conflicting information. Finally, an EA should not over-exploit the provided knowledge because it can introduce a strong bias that leads the EA to neglect exploration of the tradespace for novel solutions that are not captured by the provided knowledge, and thus cause premature convergence on local optima [169, 172].

The above issues are applicable not only to EAs that utilize expert knowledge available prior to the search, but also to *knowledge-driven optimization* (KDO) [9] algorithms, which integrate data mining into the EA to learn new knowledge from solutions discovered during the optimization. As KDO algorithms become more popular [8, 37, 65, 75, 132], there is a greater need for approaches that address the above issues on effective knowledge utilization, especially because KDO algorithms can automatically extract and apply new knowledge without a human in-the-loop to assess the utility of the extracted knowledge. Moreover, many data mining techniques used within KDO algorithms extract multiple rules or patterns that guide the optimization, and without human intervention, the KDO must decide which, if any, of the extracted knowledge to apply.

The existing KDO-compatible methods to incorporate knowledge can be largely categorized into EAs that encode knowledge in specialized evolutionary operators referred to in this paper as *knowledge-dependent operators* [75] or a special type of constraint referred to in this paper as *knowledge-dependent constraints*. Knowledge-dependent (KD) operators modify specific decision variables by biasing them towards or away from predetermined values, whereas KD constraints are used to penalize or eliminate solutions that are not consistent with the provided knowledge. The literature contains successful examples of using KD operators [13, 22, 77, 118] and KD constraints [65, 132], but there are no known experiments comparing their efficacy. Instead, developers of knowledge-intensive EAs and KDO algorithms show that an EA that leverages domain-specific knowledge has superior search performance over an analogous knowledge-independent EA. As a result, there is little guidance on which approach to use when implementing knowledge-intensive EAs or KDO algorithms.

The purpose of this paper is to address the lack of comparative experiments on methods to incorporate knowledge into EAs and to help identify which ones are promising for future knowledge-intensive EAs and KDO algorithms. To generalize the findings to cover both knowledge-intensive EAs and KDO algorithms, the scope of the experiments includes existing methods that can accommodate both knowledge available prior to the search and knowledge learned during the search. However, to eliminate confounding effects from the data mining strategies and to focus specifically on analyzing the strategies to incorporate knowledge, the EAs in the experiment do not implement any data mining. Instead, the EAs in the experiment only utilize expert knowledge available prior to the search. Consequently, some of the findings from the experiments may not be directly applicable to KDO algorithms because of differences between expert knowledge and knowledge learned through data mining algorithms. Nevertheless, we believe that the insights into how or when an EA should leverage available knowledge will also be valuable for KDO algorithm developers.

The comparative experiment presented in this paper involves benchmarking one EA using KD operators and two EAs using different implementations of KD constraints against an analogous knowledge-independent EA on a design problem for a climate-monitoring satellite system. This design problem is a multiobjective, nonlinear, and non-convex version of the general assignment problem [60], which is NP-Hard and common in the combinatorial optimization literature. The satellite system design problem is appropriate for comparing methods for incorporating knowledge because prior work [163, 165] and the literature on climate-monitoring satellite systems [102] allow us to supply each knowledge-intensive EA with a rich set of domain-specific knowledge. Moreover, the results indicate that some of the supplied design heuristics are much more effective in discovering high-quality solutions for the presented problem, which reflects a realistic situation where it is difficult or impossible to assess the quality of heuristics prior to the search [22]. Therefore, while the performance of a knowledge-intensive EA will generally depend on the quality of the provided knowledge, the problem and experiment presented in this paper are suitable for comparing the methods to incorporate knowledge and identifying the strengths and weaknesses in how each approach utilizes the provided knowledge to improve an EA's search performance.

The remainder of this paper is organized as follows. Section 3.2 explains the algorithms used in the experiments. Section 3.3 and Section 3.4 introduce the design problem and the available knowledge that are provided to each knowledge-intensive EA. Section 3.5 gives the details on the experimental setup. Section 3.6 provides the results and Section 3.7 discusses the conclusions and future work.

## 3.2 Method

In this paper, we empirically compare the efficacy of applying knowledge through operators and constraints. The purpose of the experiment is twofold. First, it should confirm the findings from related work [77, 118] that an effective EA should adapt its use of knowledge as the search progresses. Specifically, it should be beneficial if the EA learns to differentiate knowledge that improves the search performance from that which hinders the search and modifies the search strategy accordingly. Second, the experiment should elucidate some of the advantages and disadvantages of applying knowledge through operators versus through constraints.

We compare a total of four EAs: one baseline knowledge-independent EA, one EA that applies KD operators, and two EAs that apply KD constraints. Successful knowledge-intensive EAs should attain at least the same level of solution quality as the knowledge-independent EA and discover those solutions faster. Each knowledge-intensive EA extends the baseline algorithm for a fair comparison that helps isolate the effects of applying knowledge through operators and constraints. While there are a myriad of EAs to select as the baseline algorithm, we use $\epsilon$-MOEA [38]. $\epsilon$-MOEA is well known and has theoretical convergence and diversity guarantees through its use of $\epsilon$-dominance [105] in its archive, which stores the best solutions discovered so far. Moreover, it performs well on various multiobjective optimization problems [112] and has been shown to be robust to its parameterization [72]. Finally, $\epsilon$-MOEA is a steady-state algorithm that updates its population one solution at a time, which allows an adaptive search strategy to quickly react and modify its strategy. Note that while other EAs can be used as the baseline algorithm in a general implementation of a knowledge-intensive EA, the strengths of $\epsilon$-MOEA make it a suitable algorithm for the comparative experiment.

All EAs are applied to the discrete-valued, multiobjective test problem described in Section 3.3. To handle the discrete-valued variables, we replace $\epsilon$-MOEA's default simulated-binary crossover and polynomial mutation operators with single-point crossover and a discrete-valued uniform mutation. The extensions of $\epsilon$-MOEA are shown in Algorithm 4, where $\gamma$ and $\omega$ are the parent and offspring solutions, respectively. The details of each knowledge-intensive EA are provided in the following subsections.

### 3.2.1 Knowledge-intensive EA: Operators

The knowledge-intensive EA employing KD operators will be referred to as O-AOS. It utilizes an AOS to adapt the use of multiple KD operators alongside knowledge-independent operators and focuses on applying the operators that lead to improving solutions. Applying KD operators with an AOS has been successfully employed within a knowledge-intensive EA [77] and a KDO algorithm [75]. O-AOS is similar to the algorithm from [77] with the same credit assignment but a modified operator selection strategy explained below. The credit assignment of an AOS defines when and how to reward operators for the solutions they create, and the operator selection strategy selects the next operator to apply based on the rewards [56]. O-AOS uses one of the recommended credit assignment strategies from [78] to reward operators for improving the quality of the archive, where an operator $o_i$ in the set of operators $O$ is rewarded one credit $c_{i,t} = 1$ at time $t$ if it creates a solution entering the archive. In the next iteration, adaptive pursuit [173] selects an operator with a probability $p_{i,t+1}$, which is updated according to equations (1)-(4). A minimum selection probability $p_{min}$ is set to 0.03 to encourage some exploration of poorly performing operators, in case they begin to perform

Algorithm 4: Extended $\epsilon$-MOEA

```
1:  Initialize population $P$ and archive $A$
2:  $t \leftarrow 0$
3:  **while** termination criteria not satisfied **do**
4:      $t++$
5:      $\gamma \leftarrow$ select parent solutions
6:      **if** O-AOS **then**
7:          $o_i \leftarrow selectOperator(\gamma)$
8:      **else**
9:          $o_i \leftarrow$ single-point crossover
10:     **end if**
11:     $\omega \leftarrow o_i.operate(\gamma)$
12:     **if** $\epsilon$-MOEA **then**
13:         $P.update(\omega)$
14:         $A.update(\omega)$
15:     **else if** O-AOS **then**
16:         $P.update(\omega)$
17:         $A.update(\omega)$
18:         Compute credit $c_{i,t}$
19:         Update selection probabilities
20:         Reward $c_{i,t}$ to $o_i$
21:     **else if** C-DNF **then**
22:         $P.update_{DNF}(\omega)$
23:         $A.update_{DNF}(\omega)$
24:     **else if** C-ACH **then**
25:         $P.update_{ACH}(\omega)$
26:         $A.update_{ACH}(\omega)$
27:     **end if**
28: **end while**
```

better as the population evolves. $\alpha$ and $\beta \in [0, 1]$ are user-specified parameters, which are set to default values of 0.8 [173] for this paper. Operator selection and credit assignment occur on line 7 and line 20, respectively, of Algorithm 4. The population and archive are updated according to the same procedures as $\epsilon$-MOEA.

$$q_{i,t+1} = (1 - \alpha) \cdot q_{i,t} + \alpha \cdot c_{i,t} \tag{3.1}$$

$$o^* = \operatorname*{argmax}_{o_i \in O} q_{i,t} \tag{3.2}$$

$$p_{max} = (1 - (|O| - 1) \cdot p_{min}) \tag{3.3}$$

$$p_{i,t+1} = \begin{cases} p_{i,t} + \beta \cdot (p_{max} - p_{i,t}) & \text{if } o_i = o^* \\ p_{i,t} + \beta \cdot (p_{min} - p_{i,t}) & \text{otherwise} \end{cases} \tag{3.4}$$

Algorithm 5 shows one modification applied to the operator selection strategy, where a KD operator is applied only if it can use its encoded knowledge to modify the selected parent solutions. Many KD operators are encoded in the form of an "if-then" rule where a solution must meet a condition in order for the knowledge to be applied [22,75,77]. With a normal operator selection strategy, a parent solution that does not satisfy the KD operator's condition would remain unmodified but would be evaluated, wasting limited evaluations. Therefore, if a KD operator is selected but its conditions are not met by the parent solutions, then, in line 4 - 6, the operator is given 0.0 credit, the selection probabilities are updated, and a new operator is selected to modify the parent solutions. With this method, KD operators that are no longer able to operate on solutions in the population can never be selected as $o^*$ in equation (2). Typically, the condition of a KD operator is a specific pattern in the design variables or other computed values recorded after solutions are evaluated, so it is reasonable to assume that it is computationally inexpensive to determine if a solution satisfies the condition.

Algorithm 5: selectOperator($\gamma$)

```
Input: Parent solutions $\gamma$
1: $o_i \leftarrow$ select operator according to probabilities
2: for all solution $y \in \gamma$ do
3:     if $y$ does not meet $o_i$ conditions then
4:         Reward 0.0 to $o_i$
5:         Update selection probabilities
6:         return selectOperator($\gamma$)
7:     end if
8: end for
9: return $o_i$
```

### 3.2.2 Knowledge-intensive EAs: Constraints

The two knowledge-intensive EAs employing KD constraints employ the popular constrained-domination principle [34] method used in NSGA-II [39], but instead of penalizing solutions based on their constraint violations, solutions are penalized based on their knowledge violations, as shown in Algorithm 6. When comparing two solutions, the one with a smaller knowledge violation is preferred, and the consistent solution is always preferred over an inconsistent one, regardless of their objective values. In the presence of multiple KD constraints, the knowledge violations are first normalized to weight all KD constraints equally and then summed. If both solutions are consistent, regular selection methods are applied (e.g. Pareto dominance) to select the preferred solution. This process in Algorithm 6 is used for selecting parent solutions in line 5 and updating the population and archive in lines 22 - 26 of Algorithm 4. The following two knowledge-intensive EAs differ in their computation of the knowledge violations.

Algorithm 6: constrainedDomination($x, \hat{x}$)

```
Input: Solutions $x, \hat{x}$
1: $computeViolation(x, \hat{x})$
2: if both $x$ and $\hat{x}$ are consistent then
3:     return $select(x, \hat{x})$
4: else if $x$ is consistent then
5:     return $x$
6: else if $\hat{x}$ is consistent then
7:     return $\hat{x}$
8: else
9:     return $\operatorname*{argmin}_{y \in \{x, \hat{x}\}}(y.violation)$
10: end if
```

The first of the two EAs utilizing KD constraints will be referred as C-DNF and is based on LEM [132], where multiple KD constraints are combined into a disjunctive normal form (DNF) rule. The knowledge violations for solutions are computed using Algorithm 7. A solution is deemed consistent in line 7 if it satisfies the DNF rule (i.e. consistent with at least one of the KD constraints). The solutions that do not satisfy the rule will be penalized by the overall knowledge violation. While C-DNF does not adapt its search strategy to prefer the knowledge that produces more improving solutions, it is able to accommodate conflicting knowledge with the DNF since only one KD constraint needs to be satisfied. Moreover, C-DNF switches to a more conventional evolutionary search to explore the tradespace as the solutions begin to satisfy at least one of the KD constraints and are penalized less frequently.

The other EA, C-ACH is jointly inspired by the KDO algorithm from Gaur and Deb [65] and an adaptive constraint-handling method based on [185]. The knowledge violations for solutions are computed using Algorithm 8. In lines 4 - 8, Algorithm 7: $computeViolation_{DNF}(x, \hat{x})$

```
Input: Solutions $x, \hat{x}$
1: for all $y \in \{x, \hat{x}\}$ do
2:     $y.violation \leftarrow 0$
3:     for all KD constraints $kdc(y)$ do
4:         if $kdc(y) > 0$ then
5:             $y.violation+ = kdc(y)$
6:         else
7:             $y.violation \leftarrow 0$
8:             break
9:         end if
10:    end for
11: end for
```

C-ACH activates each KD constraint with a probability proportional to the number of solutions in the archive that are consistent with the constraint's knowledge. Therefore, a KD constraint that is satisfied by many high-quality solutions will be enforced more frequently. Otherwise, the knowledge is not seen as beneficial to the search and is not applied often. More than one KD constraint can be imposed simultaneously, and if two solutions have some knowledge violation, the better solution is determined based on the sum of the knowledge violations from the enforced KD constraints. C-ACH handles conflicting knowledge by biasing the application of the KD constraint that is more consistent with the solutions in the archive. In the event that the conflicting KD constraints are equally preferred, their stochastic application makes it possible, although does not guarantee, that only one of the conflicting KD constraints is enforced. As with O-AOS, a minimum probability is set for each constraint to 0.03 in line 6 to encourage the exploration of enforcing constraints that are not satisfied by many solutions in the archive.

Algorithm 8: $computeViolation_{ACH}(x, \hat{x})$

```
Input: Solutions $x, \hat{x}$
 1: $x.violation \leftarrow 0$
 2: $\hat{x}.violation \leftarrow 0$
 3: for all KD constraints $kdc(x)$ do
 4:     $\tau \leftarrow$ fraction of archive consistent with $kdc$;
 5:     $\rho \leftarrow$ randomNumber $\in [0, 1]$
 6:     if $\rho \leq \max\{\tau, 0.03\}$ then
 7:         $x.violation+ = kdc(x)$
 8:         $\hat{x}.violation+ = kdc(\hat{x})$;
 9:     end if
10: end for
```

## 3.3 Design Problem

Each EA is tasked with architecting a climate-monitoring system, where multiple spacecraft work together to obtain climate-related measurements such as atmospheric temperature, precipitation rate, ocean color, and atmospheric ozone. The objectives are to simultaneously maximize the system's scientific benefit and minimize its lifecycle cost. The scientific benefit is a function of how well the system satisfies over 370 measurement requirements provided by the World Meteorological Organization OSCAR database (www.wmo-sat.info/oscar). The lifecycle cost includes the costs to develop the instruments, fabricate the spacecraft bus components, assemble and test the spacecraft, launch the spacecraft, and operate the system for five years. All costs are estimated using the NASA Instrument Cost Model, Small Spacecraft Cost Model, and other cost models and cost estimating relationships from [181]. Further details of the scientific benefit and cost models can be found in [163, 165].

Table 3.1: Candidate Instruments

| No. | Instrument |
|---|---|
| 1 | Ocean color spectrometer |
| 2 | Aerosol polarimeter |
| 3 | Differential absorption lidar |
| 4 | Broadband radiometer |
| 5 | Cloud and precipitation radar |
| 6 | Polarimetric L-band synthetic aperture radar |
| 7 | Vegetation/ice green lidar |
| 8 | UV/VIS limb spectrometer |
| 9 | SWIR nadir spectrometer |
| 10 | SWIR-TIR hyperspectral imager |
| 11 | IR atmospheric sounder |
| 12 | Wide-swath radar altimeter |

The system is limited to a maximum of five spacecraft, each of which can host any combination of 12 of the candidate instruments recommended by the National Research Council's Earth Science Decadal Survey [139] and listed in Table 3.1. The spacecraft can be launched into any orbit listed in Table 3.2, which include common orbits for Earth observation such as polar and sun-synchronous orbits (SSO) with different local times of the ascending node (LTAN). A solution is represented with a $5 \times 13$ matrix $D$, where each row represents a potential satellite in the system. The first 12 columns of the matrix take on binary values representing instruments hosted on the spacecraft, where $D_{i,j} = 1$ means that the $i^{\text{th}}$ spacecraft hosts a copy of the $j^{\text{th}}$ instrument. The last column of the matrix takes on integer values to define the spacecraft's orbit, where $D_{i,13} = k$ means that the $i^{\text{th}}$ spacecraft flies in the $k^{\text{th}}$ orbit.

Each instrument provides different types of measurements (e.g. atmospheric temperature, ocean color, or atmospheric ozone) with different performance at-

Table 3.2: Candidate Orbits

| No. | Altitude | Inclination | LTAN |
|---|---|---|---|
| 1 | 400 km | $90°$ | - |
| 2 | 400 km | $97.0298°$ (SSO) | Morning |
| 3 | 400 km | $97.0298°$ (SSO) | Dawn-Dusk |
| 4 | 400 km | $97.0298°$ (SSO) | Afternoon |
| 5 | 600 km | $90°$ | - |
| 6 | 600 km | $97.7874°$ (SSO) | Morning |
| 7 | 600 km | $97.7874°$ (SSO) | Dawn-Dusk |
| 8 | 600 km | $97.7874°$ (SSO) | Afternoon |
| 9 | 800 km | $90°$ | - |
| 10 | 800 km | $98.6029°$ (SSO) | Morning |
| 11 | 800 km | $98.6029°$ (SSO) | Dawn-Dusk |
| 12 | 800 km | $98.6029°$ (SSO) | Afternoon |

tributes (e.g. spatial resolution, and accuracy), which affect how well measurement requirements are satisfied. Moreover, the spatial and temporal resolution of the instruments' measurements are directly impacted by their assigned orbit. The illumination conditions, determined by the selection of the orbit, can also affect the capabilities of the instruments. For example, the color of the ocean, which indicates the level of plankton photosynthesis activity, can only be measured if there is adequate lighting, discouraging flying the ocean color spectrometer in dawn-dusk orbits. Furthermore, a system can benefit from grouping certain instruments onboard the same payload to combine two or more data products from different instruments to obtain a new or enhanced data product. For example, an instrument that can measure the aerosol concentration can reduce the error of another instrument measuring ocean color by estimating the noise created by aerosols present in the atmosphere. Combining instruments in the same payload, however, can lead to nonlinear increases in development costs related not only to integrating the instruments, but also to increasing the requirements for power, data handling and storage, and communications. These instrument requirements along with orbit perturbations and illumination conditions, which are determined by the selection of the orbit, are used to automatically size the attitude control, power, and communication subsystems, among others, so increased instrument requirements directly impact the cost. The system's lifecycle cost is also heavily dependent on the choice of the launch vehicle, which is largely determined by the spacecraft's mass, volume, and assigned orbit. To reduce cost, our model tries to package spacecraft that are assigned to the same orbit into one launch vehicle, if it is compatible with the spacecraft's combined mass and dimensions.

This design problem is closely related to the general assignment problem, where the goal is to find an optimal assignment of a set of tasks to a set of agents. While the general assignment problem is NP-Hard [60], relaxation methods allow the problem to be tackled with branch and bound methods when the cost function and the constraints are linear and convex [20]. The instrument-instrument and instrument-orbit relationships discussed above make this design problem nonlinear and non-convex, which make the use of EAs appropriate for solving this problem.

## 3.4 Available Knowledge

This section introduces the knowledge provided to each knowledge-intensive EA. Some of the design heuristics focus on improving an individual spacecraft within the multi-satellite system, while others try to improve the whole system. In addition, there is some conflicting knowledge, where some heuristics suggest adding instruments to the satellite to improve the science benefit and others suggest removing them to decrease the lifecycle cost. The provided knowledge comes from prior work [161,165] and the authors' expertise, and it represents reasonable expert knowledge that could be applied to this problem. We emphasize, however, that the knowledge is comprised of design heuristics that are not absolute truths, so there will be some high-quality solutions that are not entirely consistent. Equations for knowledge violations involve summing the violations for each satellite $s$ in the system $S$ and normalizing over the number of satellites $|S|$, where $|\cdot|$ gives the cardinality of the set and $|S| \in \{0,1,2,3,4,5\}$ for the presented problem.

### 3.4.1 Spacecraft Mass

The majority of civilian Earth observing satellites in low Earth orbit launched by the U.S. weigh less than 3,000 kg [102], so spacecraft for this problem can also be expected to weigh less than 3,000 kg. Since the satellite subsystems are sized based on the instrument requirements, a KD operator removes a random instrument from each satellite in the system weighing more than 3,000 kg. The knowledge violation is defined as:

$$V_{mass} = \frac{1}{|S|}\sum_{s\in S} \max\left\{\frac{mass(s) - 3000}{mass(s)}, 0\right\} \tag{3.5}$$

where $mass(s)$ is the mass of satellite $s$.

### 3.4.2 Instrument Duty Cycle

Previous work studying a similar problem [77,165] suggested that the performance of the satellite system was limited by the duty cycle of the instruments. The instrument duty cycle for satellite $s$ is determined by the lowest of the duty cycle limited by onboard power $DC_p(s)$ and the duty cycle limited by data downlink rate $DC_{dr}(s)$. A low duty cycle can result from too many instruments sharing these limited resources and implies a poor design since the instruments are flown but not used all the time. To guide the search toward systems with higher duty cycles, a KD operator removes a random instrument from each satellite in the system whose instrument duty cycle is lower than 0.5. We do not impose a higher threshold that would discourage exploring spacecraft that carry several instruments to exploit data fusion opportunities but consequently have low duty cycles. The knowledge violation is defined as:

$$V_{dc} = \frac{1}{|S|} \sum_{s \in S} \max \left\{ \frac{0.5 - DC(s)}{0.5}, 0 \right\} \tag{3.6}$$

$$DC(s) = \min \left\{ DC_p(s), DC_{dr}(s) \right\} \tag{3.7}$$

where we assume all instruments hosted by $s$ have the same duty cycle $DC(s)$.

### 3.4.3 Launch Vehicle Packing Efficiency

The launch vehicle packing efficiency can be defined as the ratio of the volume of the launch vehicle's payload fairing to the volume of the spacecraft assigned to that launch vehicle. Alternatively, the packing efficiency can be the fraction of the launch vehicle's lifting capability that is used by the assigned spacecraft, where the lifting capability depends on the injection orbit. For this problem, we use the maximum of the two ratios as the packing efficiency. It is expected that high-quality solutions will efficiently utilize the launch vehicles assigned to each satellite, so if a launch vehicle has a low packing efficiency, there may be an opportunity to add more instruments to the satellites and increase their capabilities. In this case, a KD operator adds a new instrument at random to a satellite if its launch vehicle has a packing efficiency lower than 0.4. This relatively low threshold still allows some exploration of lower cost systems with small satellites that may not efficiently utilize any of the launch vehicles. The knowledge violation is defined as:

$$V_{pe} = \frac{1}{|S|} \sum_{s \in S} \max \left\{ \frac{0.4 - PE(s)}{0.4}, 0 \right\} \tag{3.8}$$

$$PE(s) = \max \left\{ \frac{\sum_{s' \in LV(s)} volume(s')}{volume(LV(s))}, \frac{\sum_{s' \in LV(s)} mass(s')}{lift(LV(s))} \right\} \tag{3.9}$$

where $PE(s)$ is the packing efficiency of the launch vehicle assigned to satellite $s$, $LV(s)$ is the launch vehicle assigned to $s$, $s'$ are all the satellites assigned to $LV(s)$, $volume(s)$ and $volume(LV(s))$ is the volume of $s$ and the available volume aboard $LV(s)$, $mass(s)$ is the mass of $s$, and $lift(LV(s))$ is the maximum mass $LV(s)$ can lift to the assigned orbit.

### 3.4.4 Instrument-Orbit Relationships

Remote sensing knowledge can exploit the instruments' properties to better assign the instruments to satellites flying in specific orbits. Instruments measuring atmospheric ozone provide the most value when flown in an afternoon SSO to observe the peak pollution, which occurs in the afternoon [124]. Additionally, optical instruments operating in the visible or near-infrared spectra require good illumination conditions and should be restrained from flying in dawn-dusk orbits with low-light conditions. Finally, the radars that look off-nadir should not be flown in the 400 km altitude orbits because at low altitudes, measurements off-nadir are significantly distorted by the atmosphere. With this information, a KD operator first identifies suboptimal assignments between the instruments and their designated orbits. For each suboptimal assignment, the operator randomly selects a satellite in the system with an acceptable orbit, and moves the instrument from the original satellite to the selected satellite. If there are no satellites flying in an orbit that would ameliorate a suboptimal instrument-orbit relationship, then the instrument remains in its currently assigned satellite. The knowledge violation is defined as:

$$V_{io} = \frac{1}{|S|} \sum_{s \in S} \frac{\theta(s)}{36} \tag{3.10}$$

where $\theta(s)$ is the number of suboptimal instrument-orbit assignments for satellite $s$ and 36 is the total possible number of suboptimal instrument-orbit assignments.

### 3.4.5 Synergistic Instrument Pairs

We can also utilize information about favorable instruments pairings, whose data products can be combined to obtain a new or enhanced data product and increase the scientific benefit provided by the system. The improvement in the system's capability is greatly facilitated when data products from instruments aboard the same spacecraft can be combined because it takes advantage of multiple measurements of the same region of Earth, taken at the same time, and from the same perspective. For example, as mentioned earlier, the accuracy of the ocean color spectrometer can be improved if it is flown together with a lidar that can measure the aerosol optical depth. Aerosols in the atmosphere reflect, absorb, and scatter light, affecting the spectral radiances measured by the ocean color spectrometer, but these can be corrected for by measuring the aerosol optical depth [61]. Such positive interactions are referred to as synergistic relationships [165], and we use 10 known synergistic instrument pairs. A KD operator examines each satellite in the system in a random order for missed opportunities of synergistic instrument pairings, i.e., when the satellite has one of the instruments in a synergistic pairing and the complementary instruments is assigned to another satellite. Then, for each missed synergy opportunity, the KD operator moves the latter complementary instrument to the former satellite to complete one of the synergy pairings. The knowledge violation is defined as:

$$V_{syn} = \frac{1}{|S|} \sum_{s \in S} \frac{\phi_s}{10} \tag{3.11}$$

where $\phi_s$ is the number of missed opportunities of synergistic pairings for satellite $s$.

### 3.4.6 Interfering Instrument Pairs

While some instrument pairs lead to more scientific value, there are also instrument pairings that have negative effects on engineering costs, and therefore lifecycle costs, because of the increased complexity of integrating the instruments on the same payload. For example, radars and lidars are large and require significant power for operation, so hosting a radar and lidar on the same spacecraft requires a large power subsystem, which in turn leads to larger and heavier attitude control and propulsion subsystems. These negative interactions are referred to as interfering relationships [165] and we use 10 specific interfering instrument pairs. A KD operator examines each satellite in the system in a random order for the presence of interfering instrument pairings (i.e. when both instruments in a pair are present). Then, for each interfering instrument pair present on a satellite, the operator breaks up the pair by moving one of the instruments in the pair to another satellite that does not contain the other instrument in the pair. The knowledge violation is defined as:

$$V_{int} = \frac{1}{|S|} \sum_{s \in S} \frac{\psi_s}{10} \tag{3.12}$$

where $\psi_s$ is the number of interfering pairings present in satellite $s$.

## 3.5 Experiment

All proposed methods were implemented using MOEAFramework [71], an open source Java-based library for developing multiobjective evolutionary algorithms. $\epsilon$-MOEA and the knowledge-intensive EAs use a single-point crossover with a probability of 1.0 and a discrete-valued, uniform mutation with a probability of 1/65, the inverse of the chromosome length. The $\epsilon$-values in the archive are set to 0.001 and \$1M for the science benefit and lifecycle cost, respectively, to limit the archive size to about 1,000 solutions. Each of the four EAs is applied to the problem from Section 3.3 with a limit of 5,000 function evaluations and for 30 independent trials, where the population randomly initialized in each trial.

Since the true Pareto front is not known for the presented problem, an approximate Pareto front $PF^*$ is constructed by merging all of the non-dominated solutions found at the end of each run and recording the non-dominated solutions from the merged dataset. To eliminate scaling difference between the objectives, the objectives are normalized using the ranges of the objective values from $PF^*$ before using hypervolume (HV) and inverted generational distance (IGD) to measure the search performance of each trial.

HV is the closed hypervolume created in the objective space by a reference point $z^*$ and each non-dominated solution $u$ in the Pareto front $PF$. A large HV implies that the $PF$ has good convergence and diversity. Specifically, HV is given by equation (3.13), where $vol(u, z^*)$ is the hypercube created with $u$ and $z^*$, and any point in $PF$ that is dominated by $z^*$ does not contribute to HV. $z^*$ is set at [-0.1, 1.1] for the normalized values of scientific benefit and hypervolume, respectively, to reduce the bias of HV to favor the center of $PF$ [5,86].

$$HV(PF, z^*) = volume(\bigcup_{u \in PF} vol(u, z^*)) \tag{3.13}$$

IGD, given by equations (3.14), computes the average of the minimum distance of each point in $PF^*$ to a point in $PF$, where the function $d(x, y)$ computes the Euclidean distance in the objective space between two solutions $x$ and $y$. A lower IGD indicates that $PF$ is closer to $PF^*$.

$$IGD(PF, PF^*) = \frac{\sum_{x \in PF^*} (\min_{y \in PF} d(x, y))}{|PF^*|} \tag{3.14}$$

HV and IGD values are recorded at intervals of 5 function evaluations, and statistical differences in HV and IGD between the knowledge-intensive EAs and $\epsilon$-MOEA are tested for using the Wilcoxon rank sum test with a significance level of 0.05. This non-parametric statistical test is commonly used to compare the performance between two or more MOEAs [28, 64]. In addition, we focus on analyzing the number of function evaluations required to reach a moderate-quality population instead of examining the final HV and IGD attained by each algorithm. A successful algorithm will discover a set of solutions close to the Pareto front by the end of the search, but the goal of this paper is to identify the effective knowledge-intensive EAs that reach high-quality populations faster than a knowledge-independent EA. In this experiment, a moderate-quality population is measured as more than 75% of the range of HV and less than 25% of the range of IGD, where the ranges of the HV and IGD are computed using the minimum and maximum recorded values across all 120 trials. These thresholds for HV and IGD are similar to the ones used in the analysis from Hadka et al. [72] to assess the search robustness of MOEAs under a multitude of parameterizations.

## 3.6 Results

### 3.6.1 Search performance

Figure 3.1 and Figure 3.2 show the HV and IGD attained by each EA over the number of function evaluations (NFE). The solid lines show the median value attained by each algorithm, and the bold portions of the lines indicate when the algorithm's performance is statistically significantly different to that of $\epsilon$-MOEA using the Wilcoxon rank sum test and a significance level of 0.05. The median and standard deviation of the HV and IGD values for each algorithm are also given in Table 3.3 and Table 3.4 at every 1000 NFE. Using a Wilcoxon rank sum test with 0.05 significance level, the † symbol in the tables indicates significantly better HV than $\epsilon$-MOEA. Boldface entries indicate best median value.

The HV and IGD metrics reveal that, of all the EAs in the experiment, O-AOS is the best performer. Compared to $\epsilon$-MOEA, O-AOS has a much sharper increase in HV and decrease in IGD, which indicates that O-AOS obtains a high-quality population with fewer evaluations. Moreover, O-AOS statistically outperforms $\epsilon$-MOEA in HV and IGD until about 2530 NFE and 2780 NFE, respectively, and O-AOS is never statistically outperformed by $\epsilon$-MOEA. By the end of the search, O-AOS achieves an average HV and IGD that are slightly inferior to those of $\epsilon$-MOEA, but the difference is not statistically significant, which suggests that their resulting final populations are similar in quality.

C-DNF and $\epsilon$-MOEA attain similar values for both HV and IGD until 1700 NFE, except for a few brief moments when C-DNF has statistically inferior values for IGD in the first 300 NFE. From 1700 NFE to the end of the search, the aver-

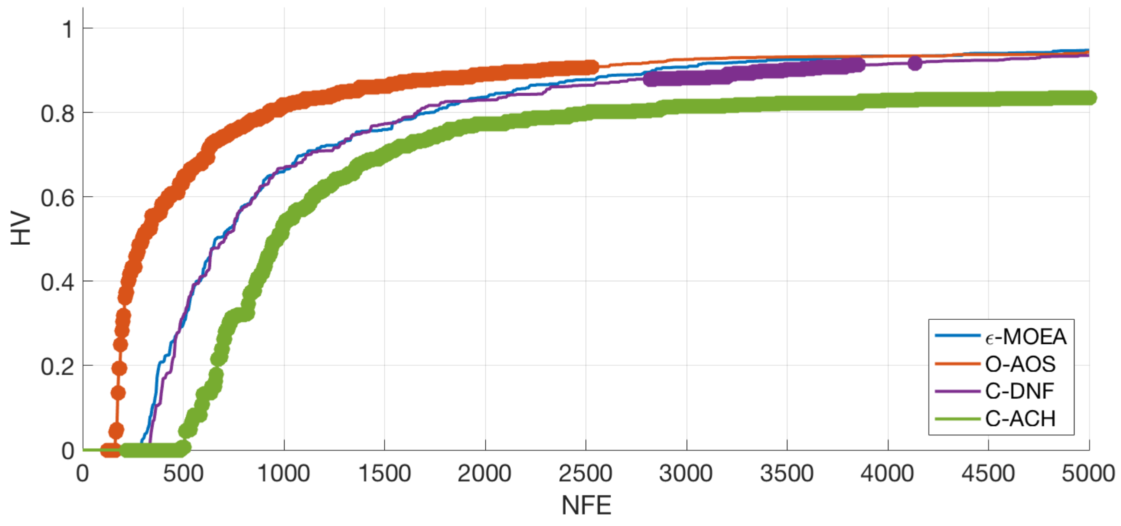

Figure 3.1: The history of the hypervolume (HV) over the number of function evaluations (NFE).

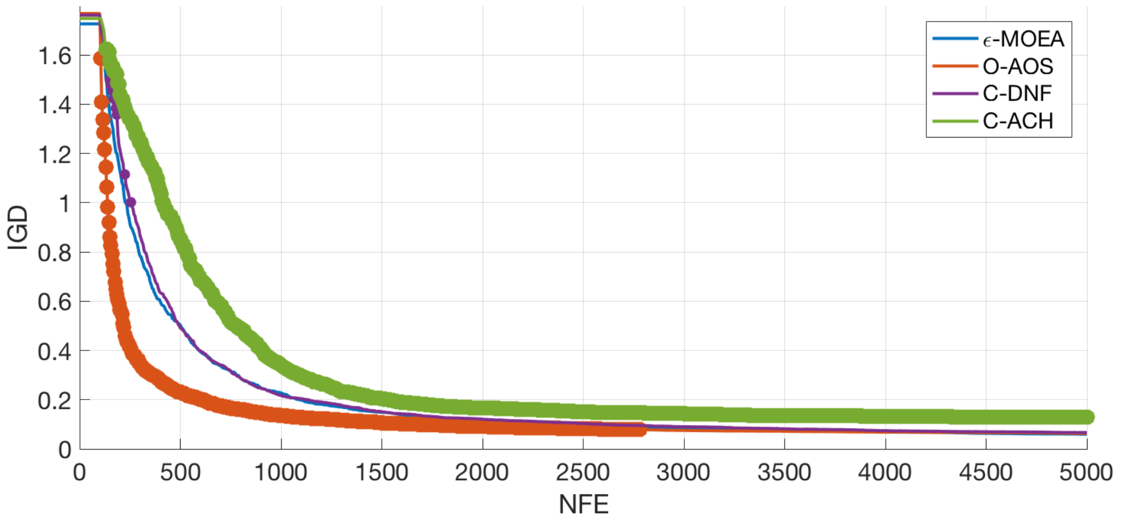

Figure 3.2: The history of the inverted generational distance (IGD) over the number of function evaluations (NFE).

age HV attained by C-DNF is slightly worse than $\epsilon$-MOEA, which is statistically significant between 2820 and 3850 NFE. For the majority of the search, however, the two algorithms have statistically equivalent performance in both HV and IGD.

Table 3.3: Median (Standard Deviation) HV values achieved by each method.

| Algorithm | 1000 NFE | 2000 NFE | 3000 NFE | 4000 NFE | 5000 NFE |
|---|---|---|---|---|---|
| $\epsilon$-MOEA | 0.6593 (0.074) | 0.8365 (0.0438) | 0.9073 (0.038) | **0.9336** (0.040) | **0.9472** (0.039) |
| O-AOS | **0.8181**$^\dagger$ (0.051) | **0.8909**$^\dagger$ (0.035) | **0.9255** (0.032) | **0.9336** (0.032) | 0.9433 (0.032) |
| C-DNF | 0.6679 (0.078) | 0.8296 (0.048) | 0.8816$^\dagger$ (0.045) | 0.9155 (0.040) | 0.9354 (0.040) |
| C-ACH | 0.5313$^\dagger$ (0.143) | 0.7732$^\dagger$ (0.055) | 0.8139$^\dagger$ (0.050) | 0.8284$^\dagger$ (0.041) | 0.8348$^\dagger$ (0.038) |

Table 3.4: Median (Standard Deviation) IGD values achieved by each method.

| Algorithm | 1000 NFE | 2000 NFE | 3000 NFE | 4000 NFE | 5000 NFE |
|---|---|---|---|---|---|
| $\epsilon$-MOEA | 0.2269 (0.048) | 0.1108 (0.025) | 0.0826 (0.019) | 0.0687 (0.020) | **0.0599** (0.019) |
| O-AOS | **0.1356**$^\dagger$ (0.030) | **0.0911**$^\dagger$ (0.020) | **0.0753** (0.019) | **0.0682** (0.019) | 0.06342 (0.018) |
| C-DNF | 0.2173 (0.049) | 0.1206 (0.029) | 0.0902 (0.024) | 0.07423 (0.020) | 0.0664 (0.020) |
| C-ACH | 0.3396$^\dagger$ (0.116) | 0.1667$^\dagger$ (0.035) | 0.1433$^\dagger$ (0.031) | 0.1330$^\dagger$ (0.028) | 0.1279$^\dagger$ (0.027) |

Finally, C-ACH is the worst performer of all the tested EAs. At every stage in the optimization, including the end of the search, C-ACH is always statistically inferior to $\epsilon$-MOEA in both HV and IGD. This indicates that compared to $\epsilon$-MOEA, C-ACH not only arrives at a significantly lower quality population at the end of the search, but also is slower in attaining a given population quality.

Figure 3.3 supplements the HV and IGD information, and it shows the set of non-dominated solutions from a representative trial for each algorithm at 500 NFE, 1,000 NFE, 2,000 NFE, and 5,000 NFE. At 500 NFE, solutions found by O-AOS are not only the closest to the approximate Pareto front $PF^*$, but also the most spread out over $PF^*$. In fact, the set of non-dominated solutions found by O-AOS strictly dominates the set of non-dominated solutions found by $\epsilon$-MOEA. In contrast, at 500 NFE, C-ACH does not find any solutions near $PF^*$. At 1,000 NFE, O-AOS maintains its superiority with it non-dominated solutions spread out along much of the length of $PF^*$. $\epsilon$-MOEA and C-DNF have found some solutions that are as good or better than some found by O-AOS, but their non-dominated sets are not as well spread. At 1,000 NFE the solutions found by C-ACH are still relatively far from $PF^*$, with some dominated by solutions found by the other three algorithms. By 2,000 NFE, much of the difference between the non-dominated solutions found by O-AOS, $\epsilon$-MOEA, and C-DNF has diminished, but O-AOS is better able to populate the low-benefit, low-cost region of the tradespace. At 2,000 NFE, C-ACH still lags behind the other algorithms, especially in the high-benefit, high-cost region. Finally, at the end of the search, the sets of non-dominated solutions found by O-AOS, $\epsilon$-MOEA, and C-DNF are comparable and cover much of $PF^*$, which is in line with their statistically equivalent HV and IGD values. C-ACH, on the other hand, struggles to discover any of the high-benefit, high-cost solutions belonging to $PF^*$.

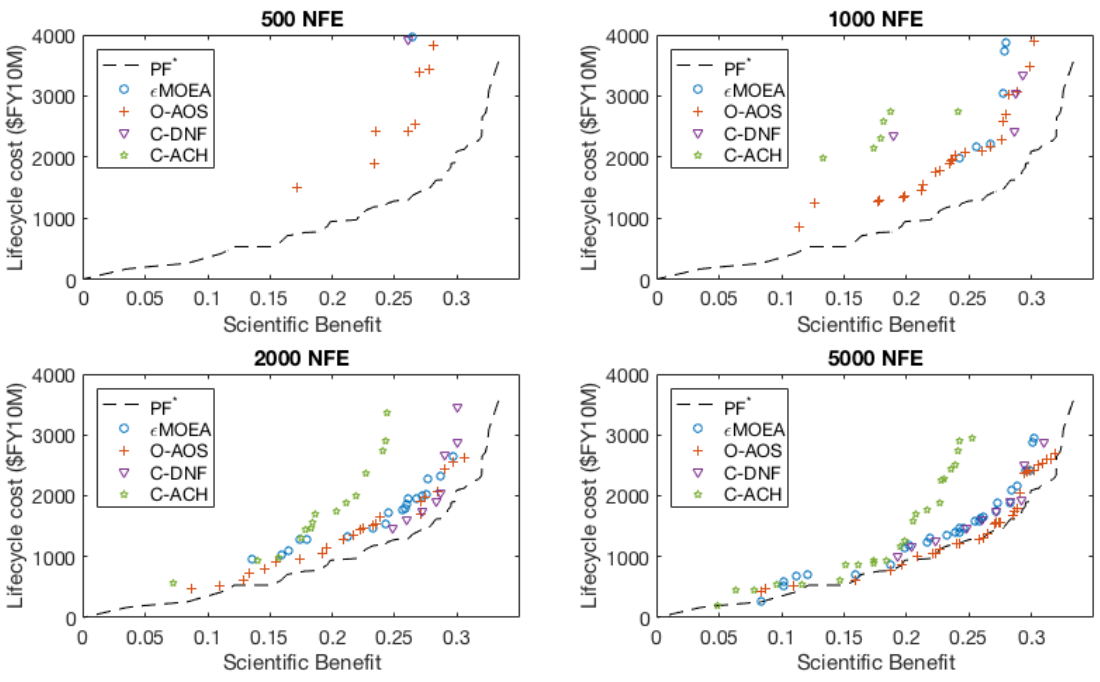

Figure 3.3: The set of non-dominated solutions found in a representative trial by each method at 500 NFE, 1,000 NFE, 2,000 NFE, and 5,000 NFE. The dashed black line shows the approximated Pareto front $PF^*$

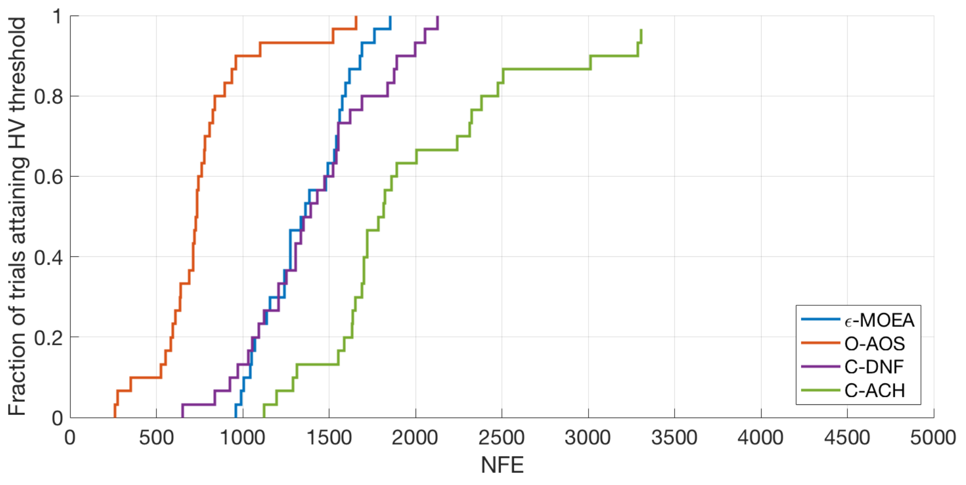

Figure 3.4: The fraction of trials by an algorithm attaining more than 75% of the range of the attained HV values as a function of NFE.

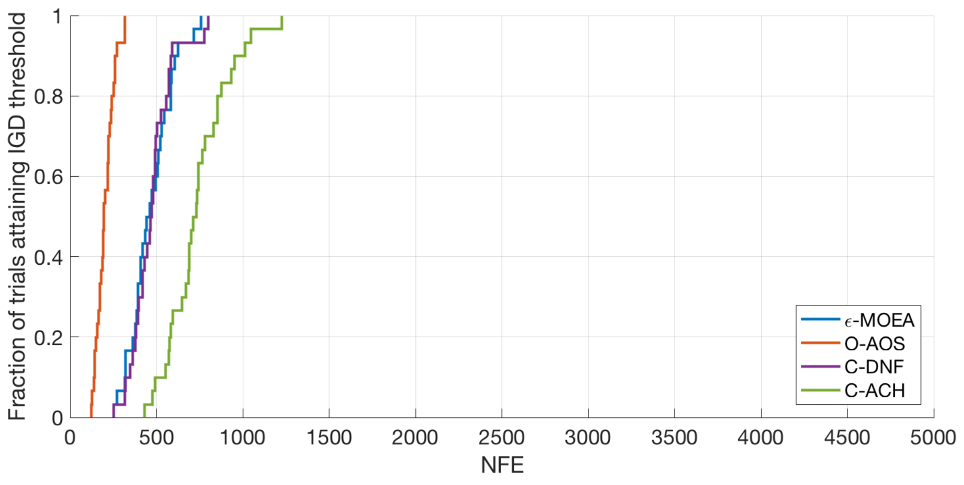

Figure 3.5: The fraction of trials by an algorithm attaining less than 25% of the range of the attained IGD values as a function of NFE.

Figure 3.1, Figure 3.2, and Figure 3.3 show that the search performance for O-AOS, C-DNF, and C-ACH compared to $\epsilon$-MEOA was better, equivalent, and worse, respectively. These differences in search performance are shown more clearly in Figure 3.4 and Figure 3.5 using the 75% and 25% threshold values for HV and IGD, which are 0.7498 and 0.5108, respectively. Compared to $\epsilon$-MOEA, O-AOS has superior search performance since 90% (27 trials) and 83% (25 trials) of its 30 trials reach the HV and IGD thresholds, respectively, with fewer NFE than the best trial of $\epsilon$-MOEA. On average, O-AOS uses 600 and 260 fewer evaluations than $\epsilon$-MOEA to reach the HV and IGD thresholds, respectively. C-DNF and $\epsilon$-MOEA have comparable search performance, as shown by their similar profiles in Figure 3.4 and Figure 3.5. Finally, 40% (12 trials) and 33% (10 trials) of the 30 trials from C-ACH reach the HV and IGD thresholds, respectively, with more NFE than the worst trial from $\epsilon$-MOEA. This corresponds to C-ACH using an average of 690 and 270 more evaluations than $\epsilon$-MOEA to reach the HV and IGD thresholds, respectively. Therefore, compared to $\epsilon$-MOEA, C-ACH has an inferior search performance.

### 3.6.2 Application of knowledge

This subsection delves into the details of how the provided knowledge helped or hindered the search in each knowledge-intensive EA. Figure 3.6 presents the knowledge violations of all solutions that entered the population across all 30 of $\epsilon$-MOEA's trials, which were not biased by any knowledge. The knowledge violations help interpret the quality or predictive power of each design heuristic, and the potential capability to improve an EA's search performance. The knowledge violations are used directly by the knowledge-intensive EAs employing KD constraints and also indicate which solutions can be modified by the respective KD operators.

The most recognizable patterns come from the knowledge violations associated with instrument duty cycle and spacecraft mass. The knowledge violations for these gradually decrease as solutions get closer to the Pareto front, and in fact, all of the solutions on the Pareto front are consistent with both KD constraints, implying that high-quality solutions tend to be small to medium-sized spacecraft. While there are several factors that determine a spacecraft's size, the main driver is the spacecraft's instrument payload. The instruments define many of the spacecraft's design requirements including electrical power, data storage and handling, communications, and attitude determination and control, among others. Fewer instruments lead to lower requirements, and therefore, smaller and lighter spacecraft components. Fewer instruments also mean that there is less competition for onboard resources such as power so the instruments' duty cycles tend to be higher, allowing each to provide more value. Note that consistency with the spacecraft mass constraint is not unique to high-quality solutions because many mediocre

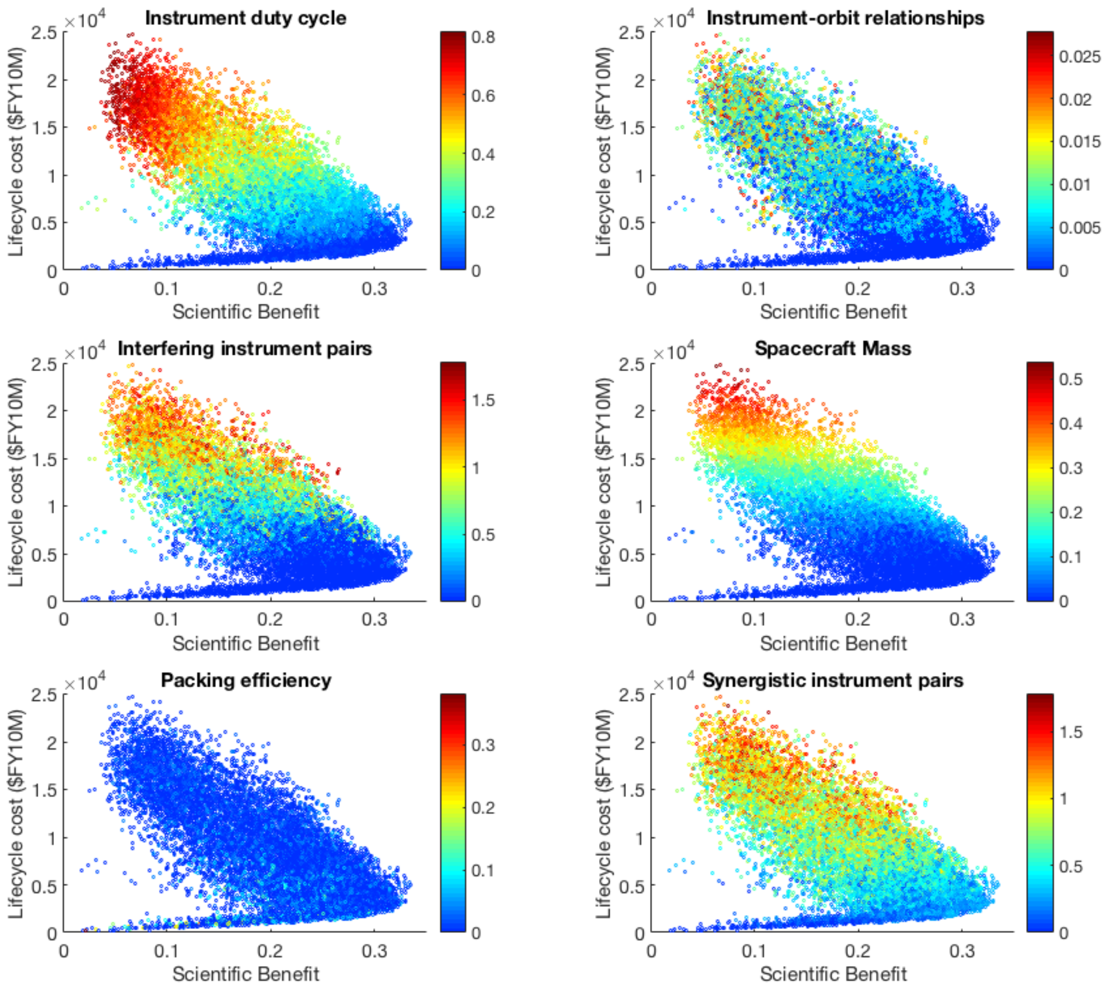

Figure 3.6: The knowledge violations corresponding to the provided knowledge for all solutions that entered the population during the optimization across all 30 trials of $\epsilon$-MOEA. The color indicates the level of each knowledge violation.

solutions are also consistent with this knowledge. Thus, the better indicator of a high-quality solution is the knowledge regarding instrument duty cycle.

The absence of interfering instrument pairs in the system and suboptimal instrument-orbit assignments are the next best indicators for solutions near the Pareto front. Since high-quality solutions tend to carry few instruments, as explained above, there are fewer chances for interfering instrument pairs to exist within a spacecraft's instrument payload. Moreover, the small number of instruments per spacecraft reduces the chances of suboptimal instrument-orbit assignments resulting from compromises between instruments on the same spacecraft that have conflicting orbit requirements. So, while the general trend is that high-quality solutions are consistent with the knowledge of interfering instrument pairs and optimal instrument-orbit relationships, the solution quality is influenced more by the number of instruments carried by the spacecraft than by the knowledge violations for interfering instrument pairs or optimal instrument-orbit relationships.

The knowledge violations for the synergistic instrument pairs follow a similar pattern, where solutions near the Pareto front tend to be more consistent with the knowledge. The low-benefit, low-cost region has little to no knowledge violation for synergistic instrument pairs because this regions contains constellations with few instruments and satellites, which reduces the chance that a pair of synergistic instruments is assigned to separate spacecraft. In contrast, many solutions on the Pareto front in the high-benefit, high-cost region are inconsistent with this knowledge. The inconsistent, high-quality solutions indicate that, for this problem, it is unfavorable to package certain instruments together to increase a satellite's capability. Taking advantage of some data fusion opportunities may not not be worth the increased costs of a larger spacecraft due to the increased requirements such as power, data storage and handling, and communications. Moreover, the addition of another instrument may be detrimental to both instruments' duty cycles, decreasing the overall scientific benefit obtained from their individual measurements and the data fusion. Finally, forcing a pair of instruments in the same payload may require one of the instruments to occupy a compromise orbit instead of an optimal one and reduce the overall scientific benefit.

Lastly, the majority of solutions have small violations regarding launch vehicle packing efficiency, indicating that 40% of the lift capability or volume of most launch vehicles is being utilized. The solutions near the Pareto front in the low-benefit, low-cost region of the objective space have larger knowledge violations with respect to launch vehicle packing efficiency. These solutions have very small satellites carrying only one instrument and assigned to high-altitude orbits (800 km altitude) to achieve good temporal resolution. These small satellites only require a fraction of the volume and lift capability of the smallest launch vehicle in our database capable of reaching high-altitude orbits.

In summary, most of the knowledge violations are small near the Pareto front, which show that the presented expertise is generally correct. The knowledge associated with the instrument duty cycle and spacecraft mass are the best indicators for high-quality solutions and are expected to be utilized heavily by the knowledge-intensive EAs. Given the relevant information above and from Figure 3.6, the following subsections describe how each knowledge-intensive EA utilized the provided knowledge to explain the search performance of each knowledge-intensive EA.

### O-AOS

Figure 3.7 and Figure 3.8 show how O-AOS rewards and selects the KD and knowledge-independent operators during the optimization. At the very beginning of the search, when O-AOS attains large gains in HV and reductions in IGD, O-AOS strongly favors the selection of the operators responsible for repairing the instrument duty cycle and spacecraft mass, both of which contain knowledge that accurately describes high-quality solutions. O-AOS does not select the other KD

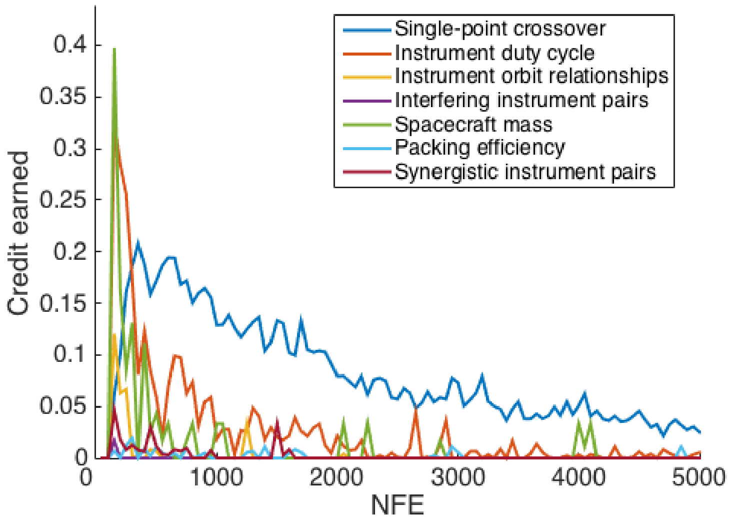

Figure 3.7: The credits earned by each operator over 5,000 NFE, averaged over the 30 trials by O-AOS.

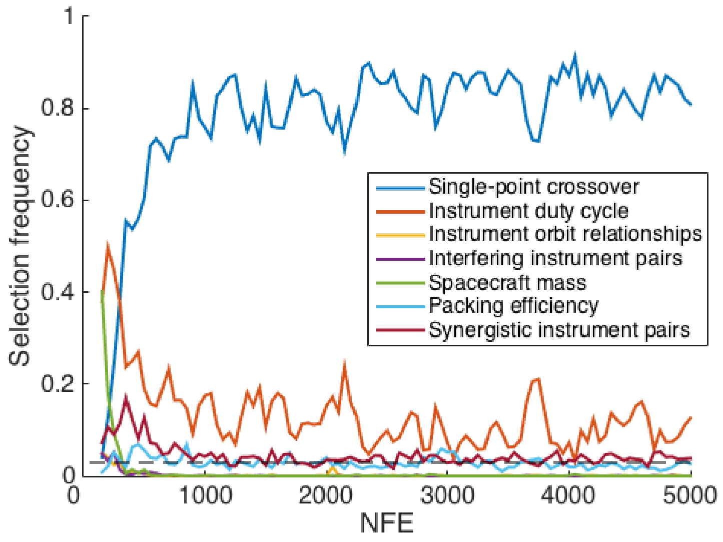

Figure 3.8: The selection frequency of each operator over 5,000 NFE, averaged over the 30 trials by O-AOS. The dashed line shows the minimum selection probability of 0.03.

operators as frequently, demonstrating O-AOS's ability to identify the most beneficial knowledge for the search.

As KD operators for instrument duty cycle and spacecraft mass remove instruments and push the search to a better region of the tradespace containing smaller spacecraft, the majority of solutions in the population become comprised of spacecraft weighing less than 3,000 kg. Once this happens, the KD operator for the spacecraft mass can no longer aid in discovering improving solutions and is rarely used for the remainder of the search, which prevents O-AOS from applying obsolete knowledge and consuming limited function evaluations. Similarly, the KD operator for duty cycle quickly becomes less successful in discovering improving solutions because many of the satellites begin to operate their instruments with a 0.5 duty cycle or greater. Unlike the spacecraft mass, however, the instrument duty cycle is affected by the illumination available in the assigned orbit for power generation, so as solutions are modified, instrument duty cycles can drop below the 0.5 threshold.

After just 400 NFE, the KD operators struggle to consistently discover improving solutions, and accordingly, O-AOS switches its search strategy to foster the exploration of the tradespace by heavily applying the knowledge-independent single-point crossover. The single-point crossover remains the most successful operator in finding improving solutions as it explores the space by recombining solutions, some of which are created by the KD operators. In contrast, the KD operators for duty cycle, synergistic instrument pairs, and packing efficiency are infrequently selected because they only contribute a few improving solutions as indicated by their credits in Figure 3.7. These ineffective KD operators continue to be selected largely due to the minimum selection probability, which consumes some of the limited function evaluations with little benefit. The overall behavior of O-AOS, however, is to focus on exploration after exhausting the benefits of the provided knowledge, demonstrating its ability to balance the exploitation of the knowledge with the exploration of the tradespace during the search.

### C-DNF

The main reason why C-DNF performs slightly worse than $\epsilon$-MOEA is C-DNF's strong preference for solutions that are consistent with at least one KD constraint. Recall that C-DNF employs the constrained-domination principle to penalize a solution only if it is inconsistent with all the KD constraints. Figure 3.9 shows that about 7.5% of the solutions entering C-DNF's population meet this criterion, most of which are created when initializing the population. In comparison, about 13% of solutions entering $\epsilon$-MOEA's population were inconsistent with all the KD constraints, and only about half of the inconsistent solutions are from the initial population. Due to the constrained-domination principle and the presence of some solutions in the population that are consistent with at least one KD constraint, C-DNF quickly discards the solutions that are inconsistent with all KD constraints and never allows such solutions to enter the population. This strict bias prevents the exploration of solutions that violate all KD constraints and possibly reduces the population diversity necessary to attain the same or better search performance as $\epsilon$-MOEA.

This bias, however, does not cause a disastrous failure in the search for C-DNF, as evidenced by the comparable HV and IGD attained over the majority of the 5,000 NFE. In fact, C-DNF and $\epsilon$-MOEA employed nearly identical search strategies on the presented problem. The vast majority of the solutions maintained and encountered by C-DNF are consistent with at least one of the KD constraints, which means that C-DNF does not apply the constrained-domination principle often to leverage the provided knowledge. Without the constrained-domination

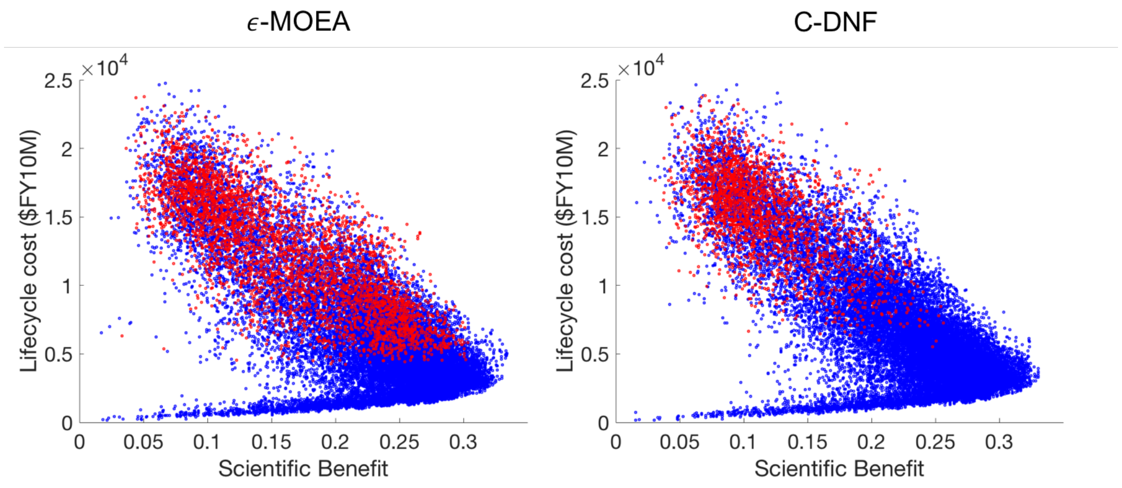

Figure 3.9: The solutions that entered the population during the 30 trials by $\epsilon$-MOEA (left) and C-DNF (right). Solutions highlighted in red are inconsistent with all KD constraints. Solutions highlighted in blue are consistent with at least one KD constraint.

principle, C-DNF and $\epsilon$-MOEA are identical. Since $\epsilon$-MOEA also maintained a population that was mostly comprised of solutions consistent with at least one of the KD constraints, the two EAs performed similarly.

### C-ACH

One reason for C-ACH's poor search performance is that it does not exploit the most beneficial knowledge early in the search. Figure 3.10 shows the fraction of the archival solutions that are consistent with each KD constraint, averaged over the 30 trials, and the fractions directly correspond to the probability of applying the respective KD constraints. C-ACH doesn't apply the knowledge for the instrument duty cycle very frequently until more than 1,000 NFE, but from Figure 3.6 we know that the instrument duty cycle is a good indicator of high-quality solutions and was heavily utilized by O-AOS to improve its search performance. The KD

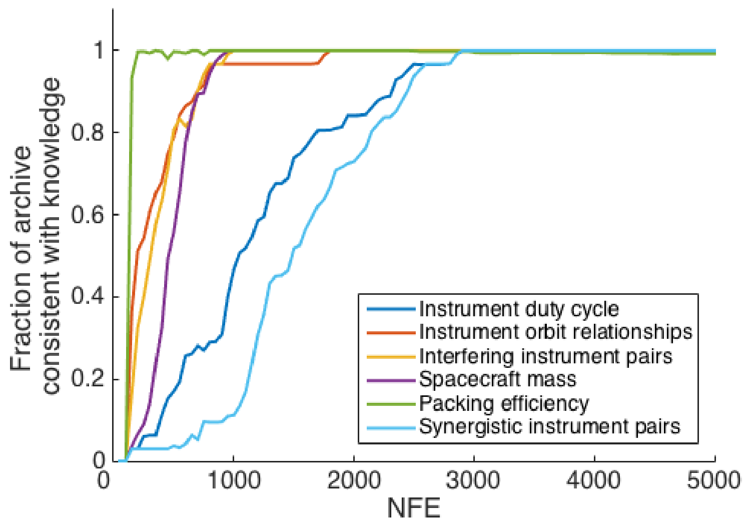

Figure 3.10: The fraction of the solutions in the archive that are consistent with each KD constraint, averaged over the 30 trials by C-ACH.

constraint for the instrument duty cycle is initially difficult to satisfy because both a spacecraft's instruments and orbit both affect the duty cycle as well as the scientific benefit and lifecycle cost. Therefore, C-ACH struggles to inject solutions consistent with the instrument duty cycle knowledge into its archive early in the search, which results in the infrequent enforcement of that KD constraint.

Another, more detrimental, cause of C-ACH's poor search performance, is its over-exploitation of the provided knowledge and strong preference for solutions that are consistent with all the KD constraints. C-DNF's preference for consistent solutions did not significantly impair the search since only one KD constraint needed to be satisfied, but in C-ACH, more than one KD constraint can cause a solution to be inconsistent. Figure 3.10 shows evidence of the strong preference for consistent solutions, where after 3,000 NFE, all solutions in the archive no longer have any knowledge violations. In other words, after 3,000 NFE, C-ACH enforces each KD constraint at every iteration and disallows solutions with any knowledge

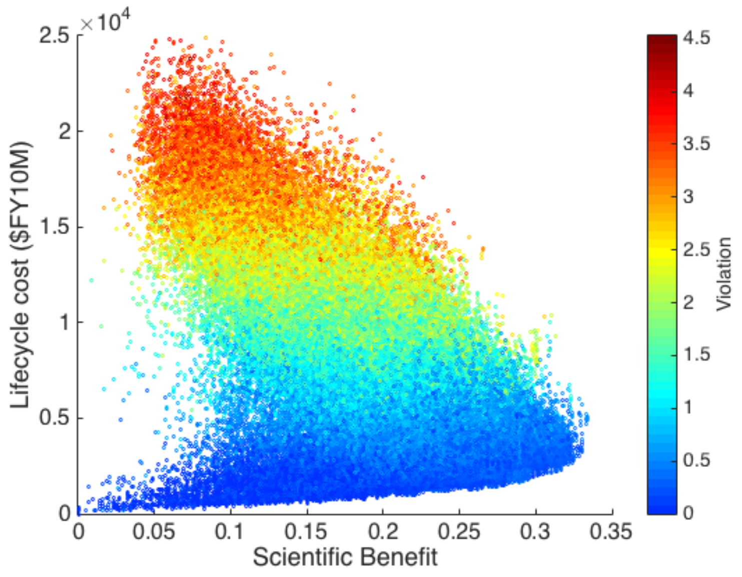

Figure 3.11: The sum of knowledge violations of solutions that entered the EA's population across all 120 trials. The color indicates the level of the overall knowledge violation.

violations to enter the population or archive. The strict preference hinders the search because many of the solutions near the Pareto front have some knowledge violations as shown in Figure 3.11. Many high-benefit, high-cost solutions are inconsistent with at least one KD constraint, but C-ACH biases the search away from these solutions and toward the low-benefit, low-costs solutions on the Pareto front. This is reflected in the non-dominated solutions found by C-ACH at 5,000 NFE, shown in Figure 3.3, since there is a lack of solutions in the high-benefit, high-cost region of the Pareto front.

## 3.7 Conclusion

Incorporating knowledge into an EA is a common approach to improving its search efficiency to obtain a high-quality set of solutions with fewer function evaluations.

Past examples of successful knowledge-intensive EAs utilize knowledge-dependent operators or knowledge-dependent constraints, but no previous work has compared the efficacy of these approaches. This paper, for the first time, conducted a comparative experiment to elucidate the strengths and weaknesses of incorporating knowledge through knowledge-dependent operators and knowledge-dependent constraints. The insights provided through the experiment will serve as a reference for algorithm developers when creating future knowledge-intensive EAs.

We experimented with three EAs that 1) utilize an adaptive operator selector to control the application of several knowledge-dependent operators (O-AOS), 2) prefer solutions that are consistent with at least one of the provided knowledge-dependent constraints (C-DNF), and 3) apply knowledge-dependent constraints with a probability proportional to the number of consistent archival solutions (C-ACH). Similar methods appear in knowledge-driven optimization algorithms, which combine data mining within an EA. Going forward, it is important to understand each methods' advantages and disadvantages in applying expert or data-mined knowledge as knowledge-driven optimization algorithms become more popular. The three knowledge-intensive EAs were benchmarked against an analogous knowledge-independent EA ($\epsilon$-MOEA) on a design problem for a climate-monitoring satellite system. Six design heuristics based on a rich set of domain knowledge and problem specific expertise were provided to the knowledge-intensive EAs. Some of the heuristics were more effective than others in identifying or producing high-quality solutions on this problem. This created a more realistic situation, where, prior to the search, it is difficult or impossible to assess a heuristic's ability to guide the optimization toward promising solutions.

Knowledge-dependent operators have dominated the literature on knowledge-intensive EAs, and our experiment showed that O-AOS successfully employed its adaptive operator selector to significantly reduce the number of function evaluations required to obtain a high-quality population. The adaptive operator selector can not only handle multiple knowledge-dependent operators encoding different sources of information, but also adapt its search strategy to focus on the most effective operators at any given stage in the search. When knowledge-independent operators, such as single-point crossover, are used alongside the knowledge-dependent operators, the adaptive search strategy helps to balance the exploitation of the provided knowledge with the exploration of other novel solutions in the tradespace. One downside of the adaptive operator selector is that it is forced to apply poorly performing knowledge-dependent operators with a small probability even toward the end of the search, consuming some function evaluations on unpromising solutions. Improvements might be attained if knowledge-dependent operators are selected at a much lower minimum probability than knowledge-independent ones to discourage using the knowledge-dependent operators once they become ineffective in producing improving solutions.

C-DNF had comparable search performance to the knowledge-independent EA on the presented problem, both in terms of the final population quality and the number of function evaluations required to reach a high-quality population. The majority of the solutions encountered during the search on the presented problem did not violate all the knowledge-dependent constraints, so the knowledge-intensive EA did not have many opportunities to apply its penalties to solutions. In other words, the provided knowledge was underutilized and the search strategy resembled that of the knowledge-independent EA, causing the two algorithms to have comparable search performance.

C-ACH was effective at eliminating inconsistent solutions from the population and archive, but this strong preference for consistent solutions overexploited the provided knowledge and prevented the search from discovering non-dominated solutions in the seemingly unpromising regions of the tradespace. Moreover, the adaptive constraint handling method was not able to focus on applying the effective knowledge about the instrument duty cycle at the beginning of the search to reach a high-quality population with few function evaluations. The combined shortcomings of incorporating knowledge with an adaptive constraint handling method resulted in significantly inferior search performance compared to the knowledge-independent EA.

The comparative experiment presented in this paper confirms that incorporating domain-specific knowledge can significantly improve an EA's search performance, but its benefits to the search efficiency heavily relies on how the EA uses the given knowledge. One limitation of our study is that the presented knowledge-intensive methods were only incorporated into $\epsilon$-MOEA. $\epsilon$-MOEA was an appropriate algorithm for the experiments since it is well-known and performs well on many multiobjective problems, but other EAs, when augmented with knowledge, may experience different search behaviors due to differences in the selection operation and population maintenance. However, the insights from the experiment should be beneficial in guiding algorithm developers to leverage the presented advantages of knowledge-dependent operators and avoid the pitfalls of knowledge-dependent constraints. The findings are also limited by the application of the three knowledge-intensive algorithms to only one real-world design problem. There may be other problems when knowledge-dependent constraints are appropriate and when knowledge-dependent operators are inappropriate. Moreover, the efficacy of each method is likely sensitive to the quality or broader characteristics of the knowledge provided. While this paper does not investigate the sensitivity of the methods to the problem or to the provided knowledge, the presented problem and knowledge were representative of a realistic implementation of a knowledge-intensive EA. Reasonable knowledge was provided to the knowledge-intensive EAs, but some of the provided heuristics were ineffective at guiding the search. Therefore, the presented problem was an appropriate benchmarking problem to reveal some of the limitations and disadvantages of methods employing knowledge-dependent operators and constraints. These limitations need to be addressed as we seek to improve the efficiency of EAs by incorporating domain-specific knowledge or develop new knowledge-driven optimization algorithms.

In addition to generalizing the findings of this paper across more problems in other domains, future work should expand on the presented work by delving deeper into analyzing the sensitivities of knowledge-intensive EAs to the quality of the provided knowledge, such as specificity or generality. A deeper understanding of these sensitivities may allow experts to better select appropriate knowledge to incorporate into EAs or help knowledge-driven optimization algorithms to downselect the more relevant information to apply during the search from the myriad of rules extracted during data mining.

CHAPTER 4

# EXTRACTING AND APPLYING KNOWLEDGE WITH ADAPTIVE KNOWLEDGE-DRIVEN OPTIMIZATION TO ARCHITECT AN EARTH OBSERVING SATELLITE SYSTEM

## 4.1 Introduction

Distributed satellite systems are becoming increasingly appealing for Earth observation as the tradeoffs between monolithic and distributed systems become more apparent and are reported in the literature. A distributed satellite system can simplify each spacecraft and reduce the system's lifecycle cost [47, 143] and provide more capabilities not attainable with a monolithic system such as improved temporal, spatial, and angular sampling [29, 135, 136]. Designing a distributed satellite system, however, presents considerable challenges due to: 1) the large number of design variables including each satellite's orbital parameters, payload, bus, and launch vehicle and their complex, nonlinear interactions; 2) many nonlinear constraints involving packaging space, power requirements, link budgets, thermal control, and on-board data handling capabilities; 3) multiple conflicting objectives of maximizing performance while minimizing cost and risk; and 4) the presence of local optima in the tradespace.

State-of-the-art decision-support tools [54, 137, 138, 163] are helping system architects overcome these design challenges for complex space systems. These tools can explore thousands of candidate architectures and present the architect with promising ones that optimally trade the multiple system objectives such as scientific value, lifecycle cost, and reliability. Moreover, these tools provide insights into the trades that the architect must make and identify the key drivers in promising architectures, both of which are helpful when architecting a complex system [42, 198]. With a deeper understanding of the key tradeoffs and drivers, a system architect can more confidently make the critical architectural decisions that have large impacts on downstream decisions and on the system's performance, cost, risk, flexibility, and other figures of merit [42, 161].

Multiobjective evolutionary algorithms (MOEA) are a popular and powerful decision-support tool for system architects and have shown much promise in tradespace exploration and analysis on real-world design problems. Unlike single-objective optimization methods, multiobjective optimization approaches explicitly allow tradeoff analysis because they keep each objective separate and provide a set of Pareto optimal solutions as opposed to a single solution. MOEAs are popular for multiobjective optimization because they can handle the nonlinearity, non-convexity, and non-differentiable properties that are common in real-world design problems [28]. Organizations including NASA, The Aerospace Corporation, and JAXA have applied MOEAs to schedule communications on NASA's Deep Space Network [93], design satellite constellations [51, 53, 54, 137, 153], and design rocket engines [104, 180].

Despite their benefits, MOEAs tend to be computationally inefficient because they rely on evaluating many solutions before converging on a set of optimal solutions. A single evaluation of a distributed satellite mission often requires propagating the orbits of each satellite over time and computing metrics (e.g. average revisit time over regions of interest) that are typically a function of the state of the satellites, ground stations, and the system as a whole. Depending on the simulation's fidelity, it can take minutes to hours to complete, so it is impractical to evaluate many hundreds or thousands of alternative solutions, especially when there are resource or time constraints. *Knowledge-driven optimization* (KDO) [7] is one approach, among many [13, 22, 75, 118, 169, 172], to improving the efficiency of an MOEA. KDO algorithms are unique because they employ data mining during the optimization to learn good design heuristics, which are based on patterns in the design variables that are common in the high-quality solutions discovered so far [25, 37, 65, 132, 164]. KDO algorithms can leverage the extracted knowledge during the optimization by encoding it into new evolutionary operators called *knowledge-dependent operators* [75] that guide the search toward promising regions of the tradespace [65, 132] and accelerate the convergence rate. Moreover, the extracted design heuristics are valuable to system architects after the search because they provide insights into design features common in high-quality architectures.

Current KDO algorithms are able to learn multiple design heuristics, but they lack a mechanism to differentiate the heuristics that help accelerate the convergence of the optimization from those that offer little benefit in advancing the search. Since data mining algorithms generally rely on statistical methods, if the data set (i.e. previously discovered solutions) is too small, a KDO can draw inaccurate or imperfect conclusions about the design features common in high-quality solutions. Even if the identified design features correctly predict the quality of a solution, the efficacy of a knowledge-dependent operator largely depends on the state of the search [74, 75, 77]. Improper use of the knowledge-dependent operators can result in suboptimal search performance and even risks premature convergence on local optima, particularly if the extracted knowledge is over-exploited [169].

This paper presents a new KDO framework called KDO\AOS that continually adapts its search strategy to apply the knowledge, if any, that is most beneficial to the search. KDO\AOS has 3 main components; 1) the underlying MOEA to manage the population of solutions, 2) a data mining method to extract design heuristics and create new knowledge-dependent operators, and 3) an adaptive operator selection (AOS) strategy to monitor the efficacy of all operators and allocate computational resources to the ones that accelerate the search. To demonstrate the efficacy of KDO\AOS and its ability to improve the convergence rate of an MOEA, KDO\AOS is applied to a design problem for a distributed satellite system for climate-monitoring and is benchmarked against an analogous MOEA that does not implement any data mining and two KDO algorithms that resemble approaches existing in the literature: a KDO that always applies all of its knowledge-dependent operators and a KDO that applies one of its operators at random.

The remainder of this paper is organized as follows. Section 4.2 provides the details of KDO\AOS, and Section 4.3 introduces the problem used in our case study. Section 4.4 presents the experimental setup. The results from the experiment are provided in Section 4.5, and Section 4.6 discusses the conclusions and future work.

## 4.2 KDO\AOS Framework

This work proposes a new framework called KDO\AOS that brings together MOEAs, data mining methods, and AOS, and draws on the strengths of each. The MOEA allows for multiobjective optimization on nonlinear and non-convex problems and is augmented with data mining to extract and create knowledge-dependent operators during the optimization that can accelerate the convergence of the search. The AOS is responsible for balancing the use of multiple knowledge-dependent operators with knowledge-independent ones to converge on optimal solutions faster while achieving sufficient exploration of the tradespace. A flowchart of KDO\AOS is shown in Fig. 4.1, where the dashed and dashed-double-dot lines show the processes associated with the AOS and data mining, respectively. The pseudocode for KDO\AOS is detailed in Algorithm 9. Line 1-2 initializes the knowledge-independent operators $O$ provided before the search and their qualities $O_q$. Lines 3-5 initialize three separate sets of solutions, where $P$ is the population containing the current solutions, $A_{best}$ is an external archive storing the best solutions found so far, and $A_{all}$ is another archive storing the unique solutions found during the search, regardless of objective value, to serve as the dataset for knowledge discovery in line 9. Data mining is applied on $A_{all}$ when certain criteria are met in line 8 such as after the population is initialized [132] or at the end of a fixed number of function evaluations [94]. Using the knowledge extracted from the data mining, $O$ is updated in line 10 by replacing a subset of the previous operators with a set of newly created knowledge-dependent operators and reseting the operators' qualities $O_q$ in line 11. In line 13, the AOS selects the next operator $o_i$ to apply, and parent solutions $\gamma$ are selected in line 14 from $P$ and $A_{best}$ for $o_i$. In line 15, $o_i$ operates on $\gamma$ to create a new offspring solution $x^{o_i,t}$. $x^{o_i,t}$ is evaluated in line 16, and $P$, $A_{best}$, and $A_{all}$ are updated based on $x^{o_i,t}$ in lines 17-19. In accordance to the credit assignment strategy, $o_i$ is rewarded for $x^{o_i,t}$ and the qualities of the operators are updated in line 20.

For the MOEA, we utilize $\epsilon$-MOEA [38] because it performs well on a multitude of multiobjective problems [72,112] and has several nice algorithmic properties. $\epsilon$-MOEA employs $\epsilon$-dominance in its external archive $A_{best}$, which prevents deterioration (i.e., the degradation of population quality) and guarantees both convergence and diversity of the solutions over time [105]. In addition, $\epsilon$-MOEA is a steady-state MOEA (i.e., it updates the population one solution at a time), which allows

Algorithm 9: KDO\AOS Pseudocode

```
$O \leftarrow$ initializeOperators()
$O_q \leftarrow$ initializeOperatorQualities()
$P \leftarrow$ initializePopulation()
$A_{best} \leftarrow$ initializeBestArchive()
$A_{all} \leftarrow P$
iteration $t \leftarrow 0$
while Termination criteria have not been satisfied do
    if dataMiningTrigger($t$) then
        knowledge $\leftarrow$ dataMine($A_{all}$)
        $O \leftarrow$ updateOperators(knowledge)
        $O_q \leftarrow$ resetOperatorQualities()
    end if
    $t++$
    $o_i \leftarrow$ selectOperator($O, Q$)
    $\gamma \leftarrow$ selectParentSolutions($P, A_{best}, o_i$)
    $x^{o_i,t} \leftarrow o_i$.operate($\gamma$)
    evaluate($x^{o_i,t}$)
    $P \leftarrow$ updatePopulation($x^{o_i,t}$)
    $A_{best} \leftarrow$ updateBestArchive($x^{o_i,t}$)
    $A_{all} \leftarrow A_{all} \cup x^{o_i,t}$
    $O_q \leftarrow$ updateQuality($o_i, x^{o_i,t}, O_q$)
end while
```

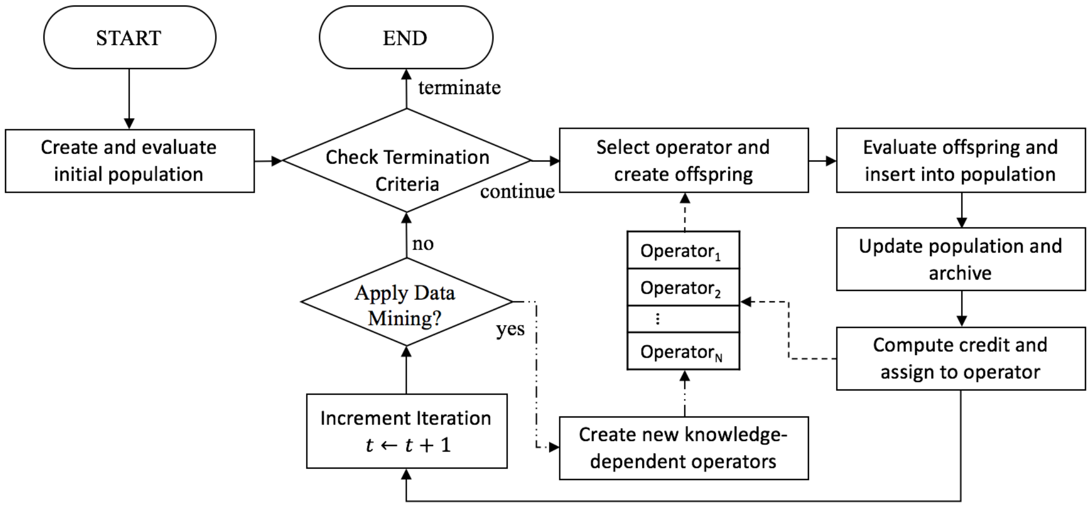
START
END
terminate
Create and evaluate initial population
Check Termination Criteria
continue
Select operator and create offspring
Evaluate offspring and insert into population
no
Operator$_1$
Operator$_2$
⋮
Operator$_N$
Update population and archive
Apply Data Mining?
yes
Compute credit and assign to operator
Increment Iteration $t \leftarrow t+1$
Create new knowledge-dependent operators

Figure 4.1: A flowchart of KDO\AOS.

for better parallelization because the population $P$ can be updated as solutions finish evaluating. To accommodate the binary decision vectors in the problem presented in Section 4.3, we make one modification by replacing the default simulated binary crossover and polynomial mutation with single-point crossover and bit-flip mutation, respectively. In KDO\AOS, these knowledge-independent operators are then utilized alongside knowledge-dependent operators created by the data mining module and controlled by the AOS.

The following subsections provide details on the data mining method and AOS strategy implemented for this paper. It should be noted, however, that KDO\AOS is a general framework and is not constrained to the implementation presented below. There are various MOEAs, data mining methods, and AOS strategies that may be more appropriate for other problems with different solution representations and can replace any of the framework's components. The choice of implementation is left to the algorithm user.

### 4.2.1 Data mining method

The objective of the data mining within KDO\AOS is to obtain a small set of general, interpretable rules that associate design features with many of the high-quality solutions near the Pareto front and few of the low-quality solutions. In addition, the obtained rules should not be redundant with each other and target different solutions near the Pareto front. In order to maintain interpretability of the extracted rules, the user first supplies the data mining method with a set of simple features that are relevant to the problem formulation. Specific examples are provided with the presented problem in Section 4.3.

The implemented data mining method consists of three steps to achieve the above goal. First, candidate rules are generated using the Classification Based on Associations (CBA) algorithm [114]. Design features that frequently appear in high-quality solutions are identified by efficiently combining a set of simple features into more complex ones that target specific regions of the tradespace. CBA does not check the specificity of the rules, however, so the resulting rules may associate design features with both low-quality and high-quality solutions. The second step applies a filter to eliminate these inaccurate rules. In the last step, a feature selection algorithm called mRMR (minimal-Redundancy-Maximal-Relevance) [147] is applied to the remaining rules to obtain the best set of non-redundant rules that will be converted into knowledge-dependent operators. The knowledge-dependent operators encoding the remaining rules modify parent solutions such that the offspring solution exhibits the good design feature. The details of the three steps are provided below.

The CBA algorithm extracts rules of the form $A \rightarrow B$, where $A$ is an attribute of an observation within a data set that is associated with the class label $B$. In the context of KDO\AOS and architecture design problems, $A$ is a set of design features that an architecture must have and $B$ is a label that defines the quality of the architecture (e.g. being non-dominated). For instance, given a set of high-performing Earth observation systems, we can extract rules such as "if a system 'employs a UV/VNIR chemistry spectrometer flying in a sun-synchronous afternoon orbit', then 'the system is close to the Pareto front'". This rule captures the spectrometer's ideal orbit for measuring the peak air-pollution during the afternoon [124]. To generate similar, potentially interesting rules, the CBA algorithm relies on an efficient rule generating algorithm called the Apriori algorithm from association rule mining [4]. Given a candidate set of potentially interesting features $F_1$, the Apriori algorithm takes a bottom-up approach and begins combining single features $A_i, A_j \in F_1$ to generate a more complex feature or feature-set such as $(A_i \wedge A_j) \in F_2$, where $\wedge$ denotes a conjunction and $F_2$ is the set of feature-sets containing two features. Since the number of possible feature-sets grows combinatorially with the size of each feature-set, a metric called support is used to prevent generating uninteresting rules. The support of a rule $A \rightarrow B$ is the proportion of a set of solutions $S$ that contain a feature or set of features $A$ and are also labeled with $B$. Specifically, support is defined as:

$$supp(A \rightarrow B) = \frac{|S_A \cap S_B|}{|S|} \tag{4.1}$$

where $|\cdot|$ is the cardinality of a set, $S_A \subseteq S$ is the set of solutions with feature A, and $S_B \subseteq S$ is the set of solutions labeled as $B$. A low support implies that the rule is not interesting because it does not apply to many solutions in the data set. Therefore, any rules that do not exceed a user-specified support threshold are ignored, and because of the anti-monotonicity property of support [3], any extension of a rule $(A_i \wedge A_j \rightarrow B)$ is also eliminated from consideration if any of its components $(A_i \rightarrow B$ or $A_j \rightarrow B)$ do not exceed the support threshold.

This special property of support allows for an efficient process to generate rules by extending only the feature-sets in $F_i$, each which contain $i$ features and surpass the support threshold, to create $F_{i+1}$.

A rule with a large enough support, however, does not necessarily mean that its feature-set is specific to a class label. If a feature-set of a rule applies to many solutions in the data set, regardless of class label, then the rule may pass the support threshold but will not be meaningful. For example, a rule that says "if a satellite contains a power subsystem, then it is a good design" would have high support, but since all satellites, including poor designs, require some power subsystem, this rule is not specific enough to high-quality solutions and thus, does not help interpret the results. To eliminate such rules, all rules produced by the CBA algorithm's rule generator are filtered based on another metric called confidence, which measures the specificity of a rule $A \rightarrow B$ and is defined as:

$$conf(A \rightarrow B) = \frac{|S_A \cap S_B|}{|S_A|} \tag{4.2}$$

A rule with a high confidence value close to 1.0 can be interpreted as "having the feature $A$ is a quasi-sufficient condition for belonging to the class $B$". However, rules surpassing a user-specified confidence threshold can be overly specific and may not relate to many of the high-quality solutions in $S_B$. Overly specific rules are unable to explain many of the high-quality solutions and also may be overfitting the data set, which does not give good predictive power on other solutions not in the data set.

Any overly specific rules are eliminated in the final step in which we apply mRMR to select a few rules from those remaining after the confidence filter that best explain as many of the high-quality solutions as possible and that are also not redundant with each other. mRMR obtains a user-specified number of $n$

rules that not only maximizes the relevance between a design feature $A$ that an architecture must have and the solution class label $B$ that defines the quality of the architecture, but also minimizes the redundancy among the $n$ selected rules. Relevance and redundancy are characterized in terms of the mutual information $I(x, y)$, given with Equation (4.3), between two random variables $x$ and $y$.

$$I(x,y) = \iint p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{4.3}$$

Relying on mutual information, mRMR greedily and incrementally selects $n$ features from a set of candidate feature-sets $\Phi$, using the following strategy:

$$\Phi_i = \Phi_{i-1} \cup \left( \max_{A_i \in \Phi \setminus \Phi_{i-1}} \left[ I(A_i, B) - \frac{1}{i-1} \sum_{A_j \in \Phi_{i-1}} I(A_i, A_j) \right] \right) \tag{4.4}$$

where $A_i$ is the $i$th feature-set to be selected, and $\Phi_{i-1}$ is the $i-1$ feature-sets already selected. The first term of Equation (4.4) maximizes the dependence between $A_i$ and solutions belonging to class $B$ (i.e. relevance), and the second term minimizes the dependencies between all pairs of previously selected feature-sets (i.e. redundancy). Equation (4.3) requires the probability density or mass functions $p(x)$, $p(y)$, or $p(x, y)$, but obtaining exact probability functions for a feature-set $A_i$ or a class $B$ would be impossible without a full factorial enumeration of the tradespace. Therefore, as commonly done in mRMR, we replace the real probability distributions with the empirical probabilities of solutions in set $S$ that exhibit feature-set $A_i$ or are labeled class $B$.

### 4.2.2 Adaptive Operator Selection

The implemented credit assignment strategy rewards an operator 1.0 credit for each solution it creates that enters the external archive $A_{best}$, which contains the best solutions found so far by the MOEA. Otherwise, the operator receives 0.0 credit. Since the archive in $\epsilon$-MOEA employs $\epsilon$-dominance, an operator is only rewarded if it can create new solutions that are not only non-dominated, but also different enough from the existing solutions in the archive. Therefore, operators are rewarded if they can both push convergence and promote diversity in the archive. This credit assignment strategy is a variation of a recommended strategy for multiobjective optimization problems [78] and has shown promise in previous studies [75, 77].

Operator selection is implemented with adaptive pursuit (AP) [174] because it is straightforward and has shown promise on several problems [70, 103, 175]. AP is based on the probability matching [67] strategy that selects operators with a probability proportional to their quality, but AP is greedier and asymptotically pursues the best operator with a specified maximum probability. For each credit $c_{i,t} > 0$ received by operator $o_i$ at iteration $t$, the quality $q_{i,t}$ of $o_i$ is updated according to Equation (4.5), where $\alpha \in [0, 1]$ is a user-specified parameter to control the importance of credits received in past iterations versus credits received in recent iterations. Then, AP identifies the operator with the highest quality in the set of operators $O$ and begins to asymptotically pursue it with a maximum probability $p_{max}$ defined by Equation 4.6, where $|\cdot|$ is the cardinality of the set. The selection probabilities $p_{i,t}$ for each operator are updated according to Equation (4.7), where $\beta \in [0, 1]$ is a user-specified parameter to control the rate of the pursuit. Note that a minimum selection probability $p_{min}$ is used to foster the exploration of all operators.

$$q_{i,t+1} = (1-\alpha) \cdot q_{i,t} + \alpha \cdot c_{i,t} \tag{4.5}$$

$$p_{max} = (1 - (|O| - 1) \cdot p_{min}) \tag{4.6}$$

$$p_{i,t+1} = \begin{cases} p_{i,t} + \beta \cdot (p_{max} - p_{i,t}) & \text{if } o_i = \underset{o_i \in O}{\operatorname{argmax}}\, q_{i,t} \\ p_{i,t} + \beta \cdot (p_{min} - p_{i,t}) & \text{otherwise} \end{cases} \tag{4.7}$$

Most AOS strategies maintain a static set of operators that do not change over time, and thus, do not have mechanisms to deal with removing ineffective operators or inserting new operators. One example where the AOS deals with a dynamic set of operators requires that new incoming operators are drawn from a prespecified database of operators [122], but since the knowledge-dependent operators will be created during the optimization, obtaining a prespecified set of operators is not feasible. Instead, a simple method to deal with the dynamic set of operators is to reset the AOS whenever new operators are created. This is a reasonable approach because the quality of any old operators that remain in the set relative to that of new operators cannot be known before the new operators are applied to the problem. Therefore, after the insertion of new operators, the AOS will reset the selection probabilities of all operators in $O$ to $1/|O|$.

## 4.3 Case Study Problem

To demonstrate the efficacy of the proposed KDO framework, it is applied to a real-world system architecture design problem in which the goal is to architect a distributed satellite system for climate-monitoring with a minimum lifecycle cost and maximum benefit to the scientific community [163, 165]. The scientific benefit is a

Table 4.1: Example Requirements for Surface Measurements

| Measurement | Attribute | Goal | Breakthrough | Threshold |
|---|---|---|---|---|
| Air temperature | Horizontal spatial resolution | 25 km | 50 km | 100 km |
| Air temperature | Temporal resolution | 3 hr | 6 hr | 12 hr |
| Air temperature | Accuracy | 0.1 K | 0.15 K | 0.3 K |
| Wind direction | Horizontal spatial resolution | 10 km | 50 km | 500 km |
| Wind direction | Temporal resolution | 1 hr | 3 hr | 6 hr |

function of the satisfaction of over 370 climate-related measurement requirements listed in the World Meteorological Organization OSCAR database (www.wmo-sat.info/oscar). The measurements include air temperature, wind speeds, air humidity, cloud cover and atmospheric ozone among others, and the database specifies the spatial and temporal resolution and accuracy required for each measurement. Each requirement has three attainment values for goal, breakthrough, and threshold values, which determine if the requirement is fully, partially, or not satisfied. Table 4.1 shows a few example requirements used for this problem. Note that fully satisfying some of the requirements is challenging with only space-based assets due to very stringent requirements such as obtaining wind direction at the surface with a temporal resolution of less than 1 hour.

To satisfy the requirements, we are interested in allocating 12 of the instruments recommended in the 2007 National Research Council's Earth Science Decadal Survey [139] listed in Table 4.2 to 5 orbits commonly used in Earth observation listed in Table 4.3. For this problem, a solution is represented by a $12 \times 5$ binary matrix $D$ where $D_{ij} = 1$ if a copy of instrument $I_i$ is assigned to orbit $O_j$, and instruments assigned to an orbit represent a single satellite carrying all of the assigned instruments in its payload. This problem has a large tradespace containing

Table 4.2: Candidate instruments

| Instrument | Description |
|---|---|
| OCE_SPEC | Ocean color spectrometer |
| AERO_POL | Aerosol polarimeter |
| AERO_LID | Differential absorption lidar |
| HYP_ERB | Broadband radiometer for radiation budget |
| CPR_RAD | Cloud and precipitation radar |
| VEG_INSAR | Polarimetric L-band SAR |
| VEG_LID | Vegetation/ice green lidar |
| CHEM_UVSPEC | UV/VIS limb spectrometer |
| CHEM_SWIRSPEC | SWIR nadir spectrometer |
| HYP_IMAG | SWIR-TIR hyperspectral imager |
| HIRES_SOUND | IR atmospheric sounder |
| SAR_ALTIM | Wide-swath radar altimeter |

Table 4.3: Candidate orbits

| Orbit | Description |
|---|---|
| LEO-600-polar | Low earth orbit with polar inclination at 600km altitude |
| SSO-600-AM | Morning sun synchronous orbit at 600km altitude |
| SSO-600-DD | Dawn-dusk sun synchronous orbit at 600km altitude |
| SSO-800-PM | Afternoon sun synchronous orbit at 800km altitude |
| SSO-800-DD | Dawn-dusk sun synchronous orbit at 800km altitude |

$2^{(5*12)} = 1.15 \cdot 10^{18}$ possible solutions.

The choice of instruments in the system is critical because each instrument provides different types of measurements (e.g. atmospheric temperature or cloud cover) with different performance attributes (e.g. spatial resolution or accuracy). Other capabilities such as temporal resolution and horizontal spatial resolution are also largely determined by the instruments' assigned orbit. Moreover, the orbit constrains the available onboard power and data downlink rates because it sets the satellite's illumination conditions and communication time with ground stations. In addition to the nonlinear instrument-orbit interactions, there are nonlinear instrument-instrument interactions [165], where instruments aboard the same satellite can complement each other by combining different measurements to create new data products or undermine each other's performance by competing for limited onboard resources such as power and data processing capabilities.

The instrument-orbit assignments also have direct impacts on the system's lifecycle cost, and high-quality solutions must optimally trade scientific benefit with lifecycle costs. In addition to estimating the instruments' development and fabrication costs, we also estimate the spacecrafts' power, data handling, and pointing requirements from their instrument payloads. These requirements are used in conjunction with the orbit properties (e.g. altitude, illumination conditions, contact time with the ground station) to appropriately size the bus components including attitude determination and control; propulsion; power; communications; thermal; and structural subsystems. Costs associated with the bus, manufacturing and integration, operations, and program overhead are estimated using the NASA Instrument Cost Model, Small Satellite Cost Model and other cost estimating relationships and models provided in Wertz, Everett, and Puschell [181]. Furthermore, the mass and the volume of the satellite and the satellite's orbit largely determine the choice of the launch vehicle, and thus the associated launch costs.

In general, data mining on design problems is challenging for classification rule mining since there are exponentially more possible design features than feasible solutions. For this problem, a design feature is represented with a matrix $F$ analogous to the binary matrix for the decision variables, where each cell contains a 0, 1, or * for either a 0 or 1 to define the value of a decision variable. Therefore, there are $3^{(5*12)} = 4.24 \cdot 10^{28}$ possible design features. To limit the number of design features that the data mining algorithm must investigate, the types of features are restricted to the ones listed in Table 4.4, where $I_i$ represents a specific instrument and $O_j$ represents a specific orbit. An additional benefit in defining the structure of the acceptable features is that the extracted features remain interpretable by a human user. Each feature is described as a predicate function and is intended to capture potentially interesting relationships in the selection of instruments or the assignment of instruments to orbits. The features *Present* and *Absent* are parameterized by an instrument and simply describe whether the given instrument is present or absent from a given architecture. *InOrbit* and *NotInOrbit* are used to capture a more specific assignment of instruments to orbits that is defined by their arguments. *Together* and *Separate* are used to describe a set of two or three instruments that are assigned together in any orbit or kept separate across all orbits, respectively. *TogetherInOrbit* combines *InOrbit* and *Together* to represent the assignment of multiple instruments together in the specified orbit. *EmptyOrbit* represents an orbit with no assigned instruments. *NOrbits* and *NInstruments* respectively check if the input argument $n$ matches the number of non-empty orbits and the total number of instruments assigned to an orbit or the total system.

The features in Table 4.4 greatly improve the interpretability of a design feature not only because they describe a matrix of 0's, 1's, and *'s with predicates resembling natural language, but also due to their implicit conjunctions and disjunctions, which require algebra to express. For example, given a design feature matrix $F$, Present($I_i$) is defined as $\sum_{O_j \in O}(F_{I_i,O_j}) \geq 1$ and *EmptyOrbit($O_i$)* is defined as $\sum_{I_j \in I}(F_{I_j,O_i}) = 0$. Moreover, the implicit conjunctions and disjunctions contained in these features can simplify a long, complex logical sentence constructed from the conjunctions or disjunctions of several literals. For example Present($I_i$) is equivalent to $\exists O_j(InOrbit(O_j, I_i))$ and *EmptyOrbit($O_i$)* is equivalent to $\forall I_j(NotInOrbit(O_i, I_j))$.

Table 4.4: Design features

| Feature Name | Arguments | Description |
|---|---|---|
| Present | $I_i$ | $I_i$ is present in at least one of the orbits |
| Absent | $I_i$ | $I_i$ is absent in all orbits |
| InOrbit | $O_i, I_j$ | $I_j$ is assigned to $O_i$ |
| NotInOrbit | $O_i, I_j$ | $I_j$ is not assigned to $O_i$ |
| Together | $I_i, I_j, (I_k)$ | $I_i, I_j, (I_k)$ are assigned together in any one of the orbits |
| TogetherInOrbit | $O_i, I_j, I_k, (I_l)$ | $I_j, I_k, (I_l)$ are assigned together in $O_i$ |
| Separate | $I_i, I_j, (I_k)$ | $I_i, I_j, (I_k)$ are not assigned to the same orbit |
| EmptyOrbit | $O_i$ | No instrument is assigned in $O_i$ |
| NOrbits | $n$ | $n$ is the number of orbits with at least one instrument |
| NInstruments | $(O_i), n$ | $n$ is the number of instruments in any orbit (or in orbit $O_i$) |

The space of possible features remains large even after limiting the data mining to the features listed in Table 4.4. Using all possible combinations of arguments in the features listed in Table 4.4, there are a total of 13,314 features. When using the Apriori algorithm to pursue logical rules that combine these single features into feature sets, the possible number of rules grows exponentially with the size of the feature set. To maintain tractability during data mining, we limit the Apriori algorithm to produce feature sets containing a maximum of three features (i.e. $A_i \wedge A_j \wedge A_k$). This restriction still allows for more than $3.93 \cdot 10^{11}$ possible rules to explain the data while keeping the rules easy-to-understand and analyze for the user.

## 4.4 Experimental setup

The efficacy of KDO\AOS is demonstrated by applying it to the design problem for a distributed satellite system for climate-monitoring and benchmarking it against $\epsilon$-MOEA, which is its underlying MOEA, and two other KDO algorithms, KDO\C and KDO\R, that differ in how they utilize the extracted knowledge. If the KDO\AOS outperforms $\epsilon$-MOEA then it suggests that there is a benefit to the KDO approach to extract knowledge and apply it during the optimization process for the presented problem. KDO\C is a variation of KDO\AOS that applies all the knowledge-dependent operators on each solution instead of adaptively applying them with an AOS. This variant represents other KDO approaches such as EMO\I, which strictly enforce the extracted knowledge onto the solutions as constraints. If KDO\AOS outperforms KDO\C, then it indicates that there is a benefit to applying knowledge through a more flexible approach with an AOS. KDO\R is another variant that is given the same operators as KDO\AOS but applies a random operator at each iteration instead of using an AOS. KDO\R resembles LEM, which enforces at least one of the rules extracted from the data mining but does not differentiate the rules that accelerate convergence from those that do not. If KDO\AOS outperforms KDO\R, then it indicates that it is beneficial to identify the most beneficial knowledge from ineffective knowledge and continuously adapt the search strategy with the AOS. All algorithms were implemented with MOEAFramework [71], an open source Java library for developing MOEAs.

Each algorithm is applied to the design problem with a maximum of 5,000 function evaluations. The population size is set to 100 solutions and is randomly initialized for each trial. Since all algorithms employ a stochastic search process, each algorithm is run 30 independent times to gather enough performance data for statistical analysis. Data mining is applied every 1,000 evaluations to update the knowledge-dependent operators multiple times during the search. Non-dominated sorting and crowding distance from NSGA-II [39] are used to identify the top 25% of the solutions in the archive $A_{all}$, which are labeled as high-quality. The rest of the solutions in $A_{all}$ are labeled as poor. Classification rules are generated to describe only the solutions belonging to the high-quality class, and the support and confidence thresholds are set to 0.0625 and 0.5, respectively. A support threshold of 0.0625 means that a rule must apply to at least 25% of the high-quality solutions, and a confidence threshold of 0.75 means that more than three-quarters of the solutions with the selected feature must be labeled as a high-quality solution. Each rule can be composed of the conjunction of up to 3 design features, and the top 4 rules as ranked by mRMR are used to create 4 knowledge-dependent operators that replace any previous knowledge-dependent operators. Single-point crossover is applied with a probability of 1.0, unless the AOS adapts its probability during the search. Bit-flip mutation is applied with a probability of 1/60, where 60 is the number of decision variables.

The algorithms are compared using a common performance metric called hypervolume (HV) that measures an algorithm's ability to obtain a population $P$ that is both close to and spread across the true or approximate Pareto front $PF^*$ [28]. HV is given by:

$$HV(P, z^*) = volume\Big(\bigcup_{u \in P} vol(u, z^*)\Big) \tag{4.8}$$

HV measures the closed hypervolume created in the objective space by a reference point, $z^*$, and the union of all the objective vectors $u \in P$, where $vol(u, z^*)$ is the hypercube created with $u$ and $z^*$. Any point in $P$ that is dominated by $z^*$ does not contribute to HV, and a large HV implies that the population has good convergence and diversity. To obtain a good reference point, we first merge all of the non-dominated solutions found at the end of each trial and record the non-dominated solutions from the merged dataset as $PF^*$. Scaling differences between the objectives are eliminated by normalizing the objectives within $PF^*$ to range from $[0, 1]$. $z^*$ is then set at [-0.1, 1.1] for the normalized values of scientific benefit and life-cycle cost, respectively, to reduce the bias of HV to favor the center of $PF^*$ [5,86]. The HV is recorded at intervals of 5 function evaluations to observe the convergence rates over time, and differences in HV between algorithms are tested for statistical significance using the Wilcoxon rank sum test with a significance level of 0.05. This non-parametric statistical test is commonly used to compare the performance between two MOEAs [28].

## 4.5 Results

### 4.5.1 Search performance

Figure 4.2 shows the HV attained by each algorithm as a function of number of function evaluations (NFE). The solid lines show the mean values, the bold portions of the line indicate when an algorithm performed statistically significantly differently with respect to $\epsilon$-MOEA using the Wilcoxon rank sum test and a significance level of 0.05, and the vertical dotted lines indicate when data mining is applied. As expected, the performance of the four algorithms are largely statistically equivalent for the first 1,000 function evaluations, before any data mining is applied. Immediately after the first time the data mining is applied at 1000 function evaluations, all three KDO algorithms experience a rapid increase in HV compared to $\epsilon$-MOEA, indicating that useful knowledge is being extracted and applied to achieve accelerated convergence. A similar response can be observed after the second application of data mining at 2,000 function evaluations, but the increase in HV is much less pronounced because the search has begun to converge. Similarly, by the end of the search, there are diminishing returns on applying benefits of the third or fourth application of data mining.

While the first and second application of data mining lead to a boost in HV, the ultimate success of a KDO algorithm heavily depends on how it applies the extracted knowledge. It is apparent that KDO\C is the worst algorithm, suggesting that applying the extracted knowledge through constraints is not effective for this problem. In fact, applying the extracted knowledge as constraints leads KDO\C to perform statistically worse than $\epsilon$-MOEA after about 2,500 function evaluations. KDO\R is the second best KDO algorithm since it converges faster than $\epsilon$-MOEA

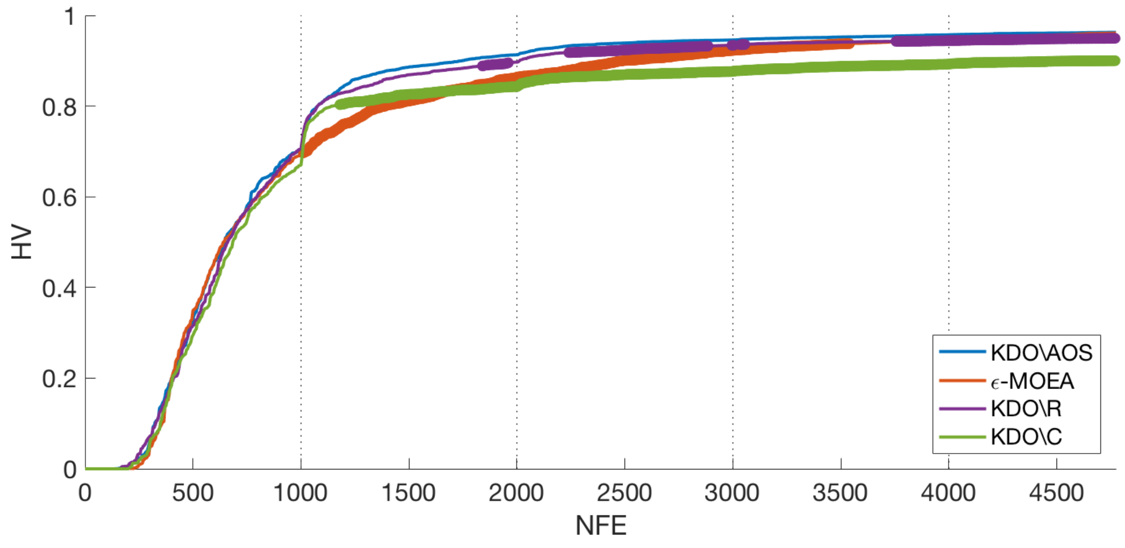

Figure 4.2: The history of the hypervolume for each algorithm

onto the approximate Pareto front $PF^*$ as evidenced by its statistically superior performance between 1,000 and 2,840 function evaluations. KDO\AOS is the best performing algorithm since it takes $\epsilon$-MOEA about 3,540 function evaluations to catch up and obtain a set of solutions with equivalent quality.

We emphasize the ability of KDO\AOS to attain a high-quality set of solutions much faster than $\epsilon$-MOEA, which is critical in cases when the function evaluations are expensive. The faster convergence rate of KDO\AOS is articulated in Fig. 4.3 that shows the sets of non-dominated solutions created from the union of solutions found by $\epsilon$-MOEA and KDO\AOS across each of their 30 trials at 1,000 NFE, 1,250 NFE, 1,500 NFE, and 2,000 NFE. As expected, at 1,000 function evaluations before any data mining, the set of non-dominated solutions found by $\epsilon$-MOEA and KDO\AOS seem comparable. By 1,250 function evaluations, however, KDO\AOS is able to find many solutions near and spread across the approximated Pareto front $PF^*$, whereas $\epsilon$-MOEA struggles to find many of the solutions in the low-cost, low-benefit region of $PF^*$. As the search progresses, both algorithms find

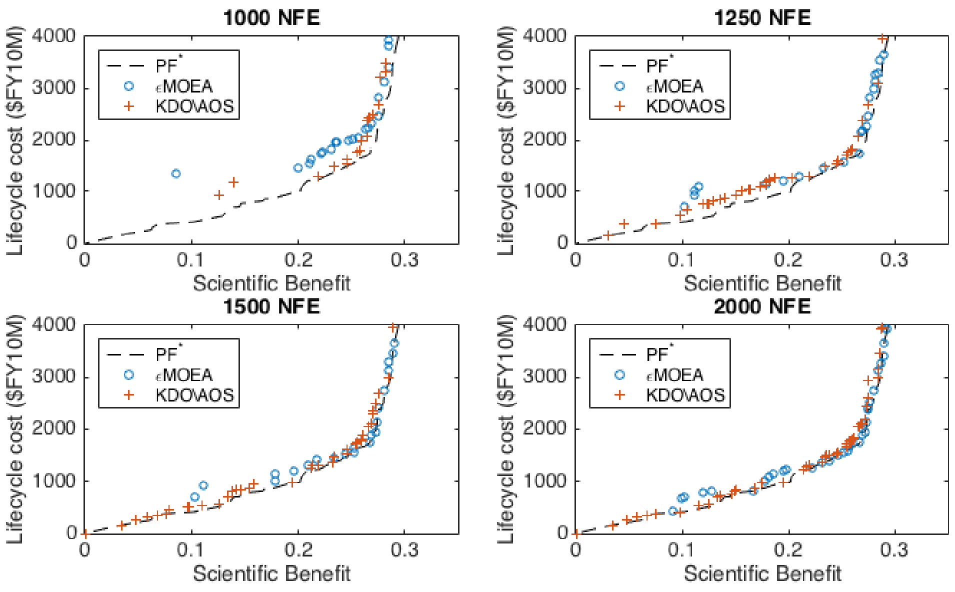

Figure 4.3: A qualitative comparison of solution quality obtained by $\epsilon$-MOEA and KDO\AOS

solutions closer to $PF^*$, but $\epsilon$-MOEA still struggles to find any solutions in the low-cost, low-benefit region of $PF^*$ across all 30 of its trials. It takes about another 500 function evaluations for the set of non-dominated solutions from 30 trials by $\epsilon$-MOEA to achieve a qualitatively similar set of solutions attained by KDO\AOS at 2,000 function evaluations.

Since the first application of data mining aided in the most significant acceleration of convergence and later applications of data mining provided less benefit as the population converged toward $PF^*$, we conducted a sensitivity analysis looking at how applying data mining early in the search affects the savings in function evaluations over $\epsilon$-MOEA to attain a high-quality set of solutions. We tried four cases where data mining was first applied at 100, 250, 500, and 1,000 function evaluations and successive data mining was still applied every 1,000 function evaluations. 30 trials were conducted for each case, and Table 4.5 shows the average number of

Table 4.5: Average number of function evaluations saved

| Algorithm | First application of data mining | 75% $HV^*$ (0.7637) | 80% $HV^*$ (0.8146) | 85% $HV^*$ (0.8656) | 90% $HV^*$ (0.9165) |
|---|---|---|---|---|---|
| KDO\AOS | 100 NFE | 370 | 415 | 490 | 220 |
| | 250 NFE | 285 | 345 | 660 | 680 |
| | 500 NFE | 350 | 480 | 605 | 940 |
| | 1000 NFE | 200 | 425 | 725 | 860 |
| KDO\R | 100 NFE | 205 | 300 | 130 | -1005 |
| | 250 NFE | 70 | 235 | 385 | 160 |
| | 500 NFE | 240 | 240 | 440 | 330 |
| | 1000 NFE | 205 | 420 | 570 | 695 |

function evaluations saved by using KDO\AOS or KDO\R instead of $\epsilon$-MOEA to reach 75%, 80%, 85%, and 90% of the HV of $PF^*$, $HV^*$. Negative numbers indicate that on average, $\epsilon$-MOEA utilized fewer function evaluations than the KDO algorithm to reach the specified HV. Values for KDO\C are omitted because of its poor performance relative to $\epsilon$-MOEA.

In general, Table 4.5 shows that there is some tradeoff in how early the first data mining should be applied and the average number of function evaluations saved to achieve a specified quality in the set of solutions. If data mining is applied right after the population is initialized like in LEM [133] or LEMMO [96], then there are few solutions in the data set to extract good features that generalize over large portions of the tradespace. So as the search progresses into unexplored regions of the tradespace, these features provide few insights into how to improve solutions and thus, should not be utilized much. On the other hand, if data mining is applied later, there is some lost opportunity in accelerating the convergence before the initial, knowledge-independent phase of the search converges to $PF^*$. KDO\AOS is more robust than KDO\R to this decision on when data mining is applied for the first time. In all but one case, KDO\AOS saves more function evaluations than KDO\R, sometimes hundreds of function evaluations, demonstrating the advantage of implementing an AOS to balance the exploitation of the extracted knowledge when it leads to improving solutions with the exploration of other regions of the tradespace as the extracted knowledge becomes less effective at improving solutions. The only time KDO\R saved more function evaluations than KDO\AOS was when data mining was applied at 1,000 function evaluations and only a moderate quality of solutions was required, 75% $HV^*$. By 1,000 function evaluations, there are enough solutions in the data set to extract design features that generalize better to the design problem, so random application of these features can lead to relatively high rates of convergence. To obtain a high quality set of solutions and the largest savings in function evaluations, however, the application of these good features must still be balanced with exploration of the tradespace as evidenced by the larger savings achieved by KDO\AOS when attaining 85% and 90% $HV^*$.

### 4.5.2 Data Mining

To understand why KDO\AOS has superior performance, this subsection presents the details on the features extracted during the optimization. Then, the next subsection describes how these extracted features are applied by the AOS to accelerate the convergence of the search.

Aggregating all the rules extracted across the 30 trials by KDO\AOS, Table 4.6 provides the relative frequency of occurrence of the features from each application of data mining. *Separate* is the most prevalent feature at every application of data mining, and its arguments tend to be either high powered instruments such as lidars and radars (AERO_LID, VEG_LID, SAR_ALTIM) or high data rate in-

Table 4.6: The relative frequency of occurrence of the extracted features

| Feature | 1000 NFE | 2000 NFE | 3000 NFE | 4000 NFE |
|---|---|---|---|---|
| Present | 0.0028 | | | 0.0028 |
| Absent | 0.1944 | 0.2361 | 0.2389 | 0.2722 |
| InOrbit | 0.0889 | 0.0750 | 0.0611 | 0.0556 |
| NotInOrbit | 0.1333 | 0.1778 | 0.1944 | 0.2333 |
| Together | 0.0056 | 0.0056 | 0.0056 | |
| TogetherInOrbit | | | | |
| Separate | 0.4944 | 0.4667 | 0.4444 | 0.4028 |
| EmptyOrbit | | | | |
| NOrbits | | | | |
| NInstruments | 0.0806 | 0.0361 | 0.0556 | 0.0333 |
| Total | 1.0000 | 1.0000 | 1.0000 | 1.0000 |

struments such as the hyperspectral imager, HYP_IMAG. The extracted *Separate* features suggest that any advantage gained by packaging these instruments on the same payload is not worth the additional cost of larger power and data handling subsystems required to operate the instruments together in the same spacecraft. Similarly, *Absent*, which was the next most frequently occurring feature at each application of data mining, is used most often with high powered and high data rate instruments including HYP_ERB, CPR_RAD, VEG_INSAR, and AERO_LID. The extracted *Absent* features suggest that the scientific benefit provided by the instruments' measurements is outweighed by the additional costs associated with hosting and operating these instruments.

The features extracted at 1,000 function evaluations provided KDO\AOS with good heuristics to significantly accelerate the convergence rate, but many of the extracted features do not generalize well over the entire tradespace. For example, $feature_{1000}$ = *Separate(HYP_ERB, VEG_INSAR)* $\wedge$ *Separate(OCE_SPEC, CPR_RAD, VEG_LID)* $\wedge$ *Separate(HYP_IMAG, SAR_ALTIM)* is a feature ex-

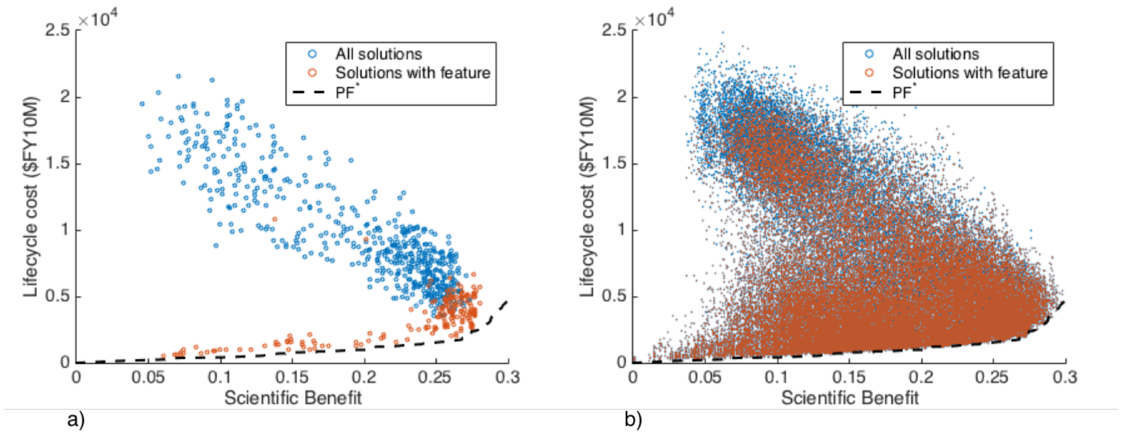

Figure 4.4: Solutions with $feature_{1000}$ from a) the first 1,000 function evaluations of one trial run and from b) every solution discovered across all 120 trials.

tracted at 1,000 function evaluations during one of the runs by KDO\AOS. Figure 4.4a shows that $feature_{1000}$ applies to many of the high-quality solutions and very few of the low-quality solutions that have been discovered so far, demonstrating that the data mining performs as expected. This feature, however, is not a good feature in general because over the entire tradespace, it does not differentiate the high-quality solutions from the poor ones as shown in Fig. 4.4b.

One reason why the features and the rules extracted at 1,000 function evaluations do not generalize well over the unexplored tradespace is because of the relatively small data set consisting of 1,000 solutions or less, which causes the rules to overfit the data. Notice in Table 4.6 that as the search progresses, there are fewer occurrences of the *Separate* features and more of the *Absent* features. The *Separate* features focus on specific instruments that should not be hosted in the same spacecraft, and this specificity is used to tease out the poor-quality solutions from the current best solutions, which may still be far from the Pareto optimal solutions. With a larger data set towards the end of the search, it becomes

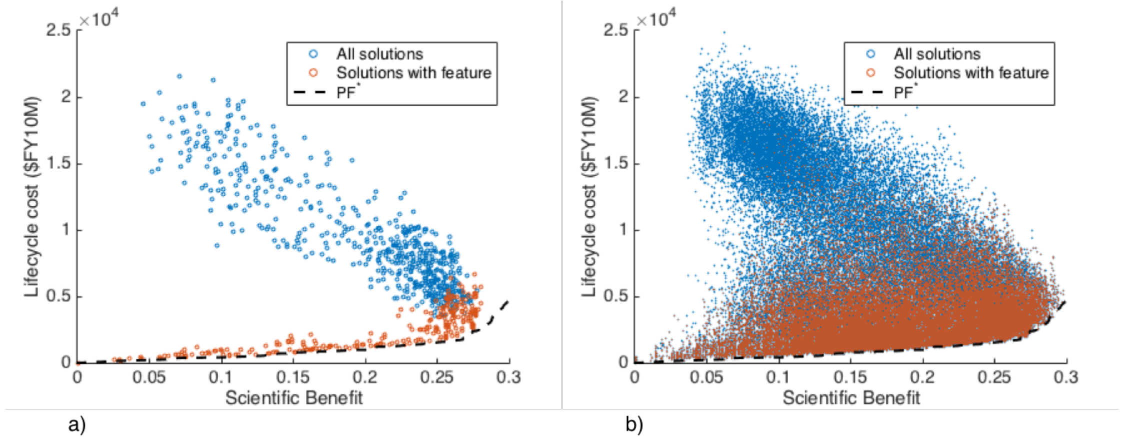

Figure 4.5: Solutions with $feature_{1000}$ from a) the first 4,000 function evaluations of one trial run and from b) every solution discovered across all 120 trials.

apparent that some of the instruments used in the *Separate* features are actually absent from the high-quality solutions, and the more general *Absent* feature helps generalize the extracted rules.

Figure 4.5 shows how the feature extracted at 4,000 function evaluations, $feature_{4000}$ = *Absent(HYP_ERB)* $\wedge$ *Separate(OCE_SPEC,CPR_RAD)* $\wedge$ *Separate(VEG_INSAR,HIRES_SOUND)*, applies to both the solutions discovered in the first 4,000 function evaluations and solutions found across all 120 trials. $feature_{4000}$ is from the same trial as $feature_{1000}$. *Separate(HYP_ERB, VEG_INSAR)* from $feature_{1000}$ is replaced with a more general *Absent(HYP_ERB)* in $feature_{4000}$ to indicate the poor value-for-cost of HYP_ERB. Additionally, *Separate(OCE_SPEC, CPR_RAD, VEG_LID)* from $feature_{1000}$ is made more general by removing VEG_LID from its arguments to become *Separate(OCE_SPEC,CPR_RAD)* in $feature_{4000}$. Comparing Fig. 4.4 and Fig. 4.5, it is clear that $feature_{4000}$ is better able to discriminate the good solutions from the poor ones even on solutions not encountered during the trial.

The inability for features extracted at the beginning of the search to clearly isolate the high-quality region of the unexplored tradespace necessitates caution when applying the features during the remainder of the search. This is why utilizing an AOS to control the application of these features can save hundreds of function evaluations over simply applying the features at random or applying the features as constraints, as seen in Table 4.5. The next subsection examines how the AOS adapted its search behavior when presented with rules that do not generalize well over large portions of the tradespace at the beginning of the search and with better rules at the end of the search.

### 4.5.3 AOS

Examining the credits earned by the operators created during the search provides further insight into the benefit of the knowledge extracted at each data mining stage. Figure 4.6 and Fig. 4.7 show the average credits earned by the operators used in KDO\AOS and their selection frequency, respectively. In both figures, the average performance of the 4 knowledge-dependent operators is presented as a representative knowledge-dependent operator since the actual operators varied during the run and between runs. The vertical dotted lines indicate when data mining is applied.

There is an overall downward trend in the credits earned by the operators, which indicates that as the search converges, it becomes more difficult to insert new solutions in the archive $A_{best}$. More interesting dynamics occur on a shorter time scale, between applications of data mining. The knowledge-dependent operators experience sudden increases in credits after each application of data mining, indicating that they are most successful in discovering improving solutions immediately after

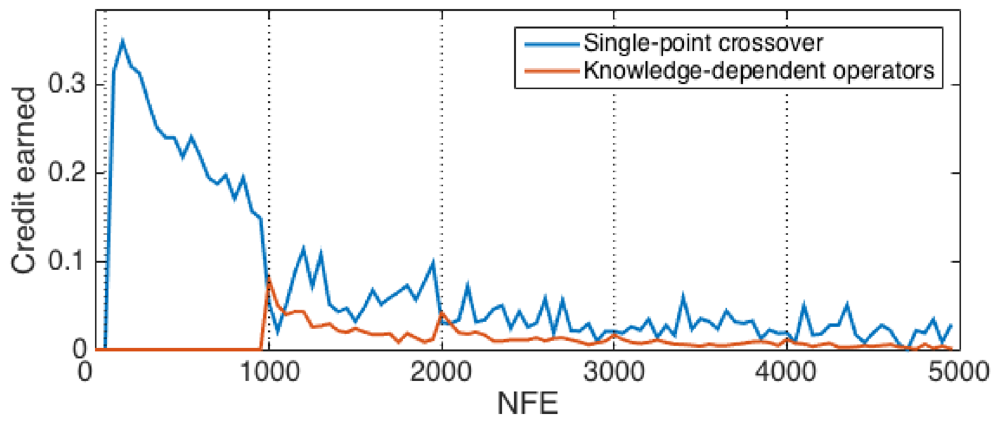

Figure 4.6: History of the credits earned by the operators used by KDO\AOS

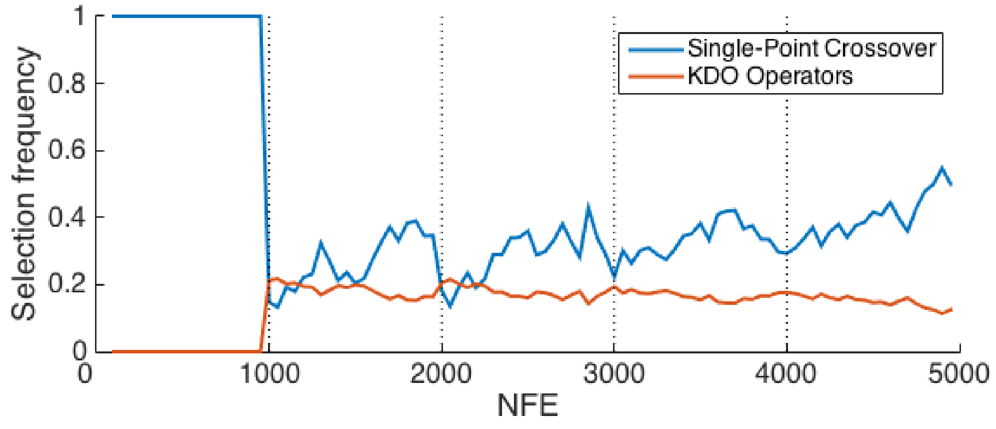

Figure 4.7: History of how KDO\AOS adapts its operator selection during the search.

extracting new rules. The good search performance of the knowledge-dependent operators is reflected in their higher selection rate by the AOS over the single-point crossover, particularly right after 1,000 and 2,000 function evaluations. The higher selection rate of the successful knowledge-dependent operators is what leads to the rapid increases in HV seen in Fig. 4.2.

As the knowledge-dependent operators impose the features onto the solutions, however, they become less effective in inserting new solutions into $A_{best}$. Several function evaluations after data mining, the knowledge-independent, single-point crossover begins to earn more credits than any of the knowledge-dependent operators, which lose efficacy over time due to their somewhat limited ability to modify solutions. The knowledge-dependent operators embody one rule that modifies solutions in a specific way, and when the rule is applied enough times, the solutions in the population can no longer be modified because they exhibit the encoded design feature or the modification can no longer create improving solutions from the current state of the population. On the other hand, the single-point crossover uses a stochastic method to modify solutions, so it can always modify a solution to explore the tradespace and discover new solutions that may enter the archive. The AOS is able to detect this change in operator performance and shifts attention from exploiting the extracted knowledge by using the knowledge-dependent operators to exploring the tradespace by frequently using the single-point crossover.

The ability to switch from knowledge-exploitation to knowledge-independent exploration is critical toward the end of the search when the extracted knowledge becomes less useful in guiding the search toward $PF^*$. Figure 4.6 shows that the knowledge-dependent operators receive very few credits after 3,000 function evaluations, but KDO\AOS quickly detects that the knowledge-dependent operators are not contributing new solutions to $A_{best}$ and focuses on selecting the single-point crossover more often than the other operators. The behavior of decreasing reliance on available knowledge and focusing on exploration as the search progresses and the knowledge becomes less beneficial is consistent with other evolutionary algorithms that leverage expert knowledge available prior to the search [77,118]. KDO\AOS's ability to detect inadequacies of the extracted knowledge and adapt its search strategy, especially at the end of the search, to balance the exploitation of the extracted information with the exploration provided by the knowledge-independent operator is the key to its superior performance over $\epsilon$-MOEA, KDO\R, and KDO\C.

## 4.6 Conclusion

This paper introduced KDO\AOS, a new knowledge-driven optimization framework, that brings together a multiobjective evolutionary algorithm, machine learning, and adaptive operator selector to more efficiently conduct multiobjective optimization on difficult real-world problems containing nonlinear and non-convex properties. Data mining augments the evolutionary algorithm by extracting design heuristics from solutions discovered during the search and creating new knowledge-dependent operators that can guide the search to achieve significantly faster convergence. The adaptive operator selector manages the application of the set of knowledge-dependent operators alongside conventional, knowledge-independent operators, and effectively balances the exploitation of knowledge with knowledge-independent exploration of the tradespace. KDO\AOS was benchmarked against a baseline evolutionary algorithm ($\epsilon$-MOEA) that did not apply any data mining, a knowledge-driven optimization algorithm that applied the knowledge-dependent operators randomly (KDO\R), and a knowledge-driven optimization algorithm that applied the extracted knowledge through constraints (KDO\C).

The results showed that KDO\AOS was the superior algorithm, saving many hundreds of function evaluations over $\epsilon$-MOEA to attain a high-quality set of solutions. Moreover, KDO\AOS saved several hundreds more function evaluations than KDO\R or KDO\C to attain a high-quality set of solutions. We showed that the adaptive operator selector was a critical component in the efficacy of KDO\AOS's search by exploiting the extracted knowledge only if and when it helped accelerate the convergence rate and by exploring the tradespace in a knowledge-independent way in the absence of effective knowledge-dependent operators.

While KDO\AOS succeeded in outperforming the other algorithms presented in this paper, there are several areas for future work for further improvements. Firstly, KDO\AOS applies data mining at fixed intervals during the search and continues to apply data mining toward the end of the search, but the results showed that knowledge extracted toward the end of the search has a greatly reduced benefit in improving the optimization efficiency or efficacy. Because the data mining is not computationally free, it may be beneficial to include a mechanism that is more adaptive and stops further data mining when the overall contribution of the knowledge-dependent operators becomes insignificant. Secondly, the rule extracted toward the beginning of the search had a limited ability to clearly distinguish high-quality solutions from mediocre ones and did not generalize well to unexplored regions of the tradespace. In this paper, we assumed that high-quality solutions have some common design feature, but in fact, the high-quality solutions are typically a diverse set of solutions and there may not be any feature that is common to all or large subsets of them. Future work will explore the benefits of identifying clusters of high-quality solutions prior to applying data mining to discover rules that target specific regions of the tradespace and better segregate high-quality solutions from mediocre ones. Lastly, the extracted knowledge from KDO\AOS only provided low-level explanations of the results, and required the authors' insights to draw more meaningful conclusions. For example, the rules containing *Separate* features only say to disaggregate a specific set of instruments from the same payload, but closer inspection was required to reveal that the majority of the *Separate* features involved the disaggregation of high-power and high data rate instruments from the same payload. Future work will investigate utilizing other design attributes of the solutions including the requirements for the payload's power and data storage.

CHAPTER 5

# CONCLUSION

## 5.1 Summary

This thesis developed a new decision support tool for multiobjective tradespace exploration that combines powerful design heuristics used by human experts to efficiently identify promising solutions with the computational power of multiobjective evolutionary algorithms (MOEA) to explore a vast region of the tradespace. The presented tradespace exploration tool relied on an adaptive mechanism to exploit the available knowledge to push the optimization toward the most promising regions of the tradespace without sacrificing the exploration of other regions of the tradespace for novel solutions not captured under conventional design considerations. Moreover, the proposed knowledge-intensive optimization algorithm can leverage multiple design heuristics, some of which may conflict with each other in how to create a high-quality solution, and modify its search strategy to focus on utilizing the knowledge that consistently leads to high-quality solutions. Finally, this thesis explored the addition of a data mining algorithm that could extract new knowledge during the optimization process by identifying common design patterns in high-quality solutions found by the optimization algorithm. This automatically extracted knowledge helps not only in guiding the remainder of the search process, but also in understanding the key driving design features that are unique to high-quality solutions. With the insights gained from the extracted knowledge and a more computationally efficient, knowledge-intensive tradespace exploration tool, system engineers and architects are provided with an opportunity to reformulate and fine-tune the design problem multiple times.

Chapter 1 discussed how the complexity of architecting a distributed spacecraft mission (DSM) motivated the development of a computationally efficient tradespace exploration tool. It examined a human-centric, expert-based approach that leverages design heuristics gained through years of experience to quickly identify promising candidate missions but is limited in its explorative capability to evaluate many different alternatives. The chapter also discussed recent efforts to utilize MOEAs to explore a larger portion of the tradespace, but these algorithms tend to be computationally inefficient since they do not leverage available knowledge about the problem or domain. Lastly, Chapter 1 proposed improving the computational efficiency of MOEAs by incorporating expert knowledge and controlling the use of that knowledge with an adaptive operator selection (AOS) strategy. The use of an AOS allows the MOEA to focus on utilizing the given knowledge that helps discover high-quality solutions, and in the event that the knowledge is not beneficial to the search, the AOS can modify the search strategy to a more conventional evolutionary algorithm that explores the tradespace.

To implement an effective AOS strategy within an MOEA, this thesis first explored the research gaps in credit assignment strategies for AOS on multiobjective problems. Chapter 2 revealed a lack of comparative studies on existing credit assignment strategies and an absence of a classification for these strategies that could guide algorithm developers. Based on the existing literature, a classification of credit assignment strategies was developed for the first time, identifying nine categories based on the type of fitness function and set of solutions used to assess an operator's impact. The classification revealed further gaps in the literature and five new credit assignment strategies were proposed to fill the gaps. This chapter also experimentally compared nine credit assignment strategies on standard benchmarking problems to evaluate their effect in elevating the generality of an

MOEA and outperforming a random operator selector.

Using the information gained about effective credit assignment strategies, Chapter 3 developed a knowledge-intensive MOEA that utilizes an AOS to control the use of knowledge-dependent operators alongside knowledge-independent operators. The performance of this knowledge-intensive MOEA was compared against two other MOEAs that incorporated knowledge through knowledge-dependent constraints, which have been utilized in the existing knowledge-driven optimization (KDO) literature. The three knowledge-intensive MOEAs were benchmarked against an analogous knowledge-independent EA on a design problem for a climate-monitoring DSM, and each EA was evaluated for its ability to attain high-quality solutions with the fewest possible number of function evaluations. In addition, each method was assessed for its ability to focus on applying the knowledge that improved the search performance, handle conflicting information that suggested improving solutions with opposing modifications, and balance the exploitation of the available knowledge with the exploration of the tradespace to prevent premature convergence on local optima. The results showed that the proposed method of utilizing an AOS to control the use of knowledge-dependent operators was the superior approach, requiring several hundred fewer function evaluations than an analogous knowledge-independent MOEA. Moreover, the results showed that the use of knowledge-dependent constraints had either no effect or a negative effect on an MOEA's search performance on the presented DSM design problem.

Chapter 4 introduced a new KDO algorithm, KDO\AOS, that augments an MOEA with a data mining algorithm and an AOS. The AOS was used to apply knowledge-dependent operators that are created from information extracted from data mining the previously discovered solutions. KDO\AOS uses both knowledge-independent operators such as single-point crossover and knowledge-dependent operators created during the optimization by reallocating computational resources to the most effective operators. The efficacy of KDO\AOS was demonstrated on a multiobjective design problem for a climate monitoring DSM and benchmarked against other KDO algorithms that apply the extracted knowledge through less adaptive methods. The results confirmed that applying new information extracted from data mining the solutions discovered during the search can significantly improve an MOEA's search performance. However, KDO algorithms that do not adaptively apply the extracted knowledge have inferior performance compared to KDO\AOS, and applying the extracted knowledge too rigidly like existing KDO algorithms may degrade the overall performance of an MOEA.

## 5.2 Main Contributions

The work in this thesis, for the first time, presents a knowledge-intensive MOEA and a KDO algorithm that utilize an AOS to control the application of knowledge-dependent operators alongside knowledge-independent ones. The main contributions and findings of this thesis are summarized below.

- A classification of credit assignment strategies for AOS on multiobjective problems was developed. This classification was based on the existing strategies in the literature and categorizes the credit assignment strategies by the type of fitness function and set of solutions used to assess an operator's impact. The classification reveals nine classes of credit assignment strategies and should be helpful for developers when defining future strategies.
- The efficacy of nine credit assignment strategies was examined by conduct-

ing a comparative experiment. Prior to the work presented in this thesis, no comparative experiment of credit assignments existed in the literature. Our comparative experiment of credit assignment strategies revealed that not all strategies are effective when compared to an operator selector that randomly chose the next operator to apply. In fact, the study showed that outperforming an MOEA that randomly applies a diverse set of operators is not trivial. The comparative study identified eight credit assignment strategies that can outperform the random operator selection approach, which can be utilized in future AOS strategies.

- This thesis developed a novel knowledge-intensive MOEA that uses an AOS to control the application of knowledge-dependent operators alongside knowledge-independent ones. Through the use of an AOS, the MOEA could incorporate knowledge from multiple sources, handle design heuristics that differed in how a solution should be improved, focus on the most effective heuristics, and balance the exploitation of the knowledge with exploration of other regions of the tradespace not captured by the knowledge. The application of knowledge through an AOS resulted in significantly improved search performance over a knowledge-independent MOEA.
- No previous work had analyzed the improvements in search performance of a knowledge-intensive MOEA when it utilized knowledge-dependent operators versus knowledge-dependent constraints. While it is generally accepted that incorporating domain- or problem-specific knowledge into an MOEA improves its search performance on a given problem, the experiments conducted in this research showed that the effect of providing knowledge to an MOEA depends greatly on how it is applied during the search. Specifically, the results showed that knowledge-dependent operators that are applied with

an AOS alongside knowledge-independent operators can achieve the best balance between exploitation of the knowledge with exploration of other regions of the tradespace not captured by the knowledge. MOEAs using knowledge-dependent constraints must be cautious when preferring solutions that are not consistent with the knowledge-dependent constraints.

- This thesis developed a novel KDO algorithm that applies knowledge extracted during the search with an AOS alongside knowledge-independent operators. The newly proposed KDO\AOS algorithm had significant improvements in search performance over a conventional MOEA that did not apply any data mining. KDO\AOS not only supplied higher-quality solutions in the same number of function evaluations, but also presented some insights into common design patterns occurring in high-quality solutions found during the optimization. These design patterns were useful in deciphering the resulting solutions of an optimization run and confirming the validity of the design choices made by the optimization algorithm in the final solutions.
- The use of an AOS within an KDO algorithm greatly improved the robustness of the search to misleading information extracted during data mining. Applying a data mining algorithm on a small sample size of solutions can result in extracting design patterns that do not generalize well to unobserved solutions. The AOS helps the KDO algorithm to detect and then ignore misleading information that does not help with the optimization.
- Through the work conducted in this thesis, an AOS code repository has been created and made public (accessible at https://github.com/seaklab/mopAOS.git). It not only allows smooth integration with MOEAFramework [71], a well-maintained Java package for experimenting and developing MOEAs, but also provides simple interfaces to develop and test new credit assignment

or operator selection strategies.

## 5.3 Discussion

This thesis has shown how an AOS can improve an MOEA in many ways. First, it can elevate the generality of an MOEA by constantly adapting the use of a diverse set of operators to the current state of the optimization. It also provides an excellent approach for balancing the exploitation of available knowledge gathered from experts or gained through data mining with the exploration of other regions of the tradespace. When knowledge-dependent and knowledge-independent operators are used by an AOS, the search is better able to cope with situations when knowledge-dependent operators are not beneficial or become ineffective in providing improving solutions. Finally, the use of an AOS allows for conflicting knowledge-dependent operators to be incorporated into the search. Many knowledge-dependent operators focus on improving one aspect of a solution, typically improving only one of multiple conflicting objectives, and each operator is likely to be effective in searching specific regions of the tradespace. Therefore, it is beneficial to be able to include a variety of knowledge-dependent operators, despite their different, perhaps contradictory, approaches to improving a solutions.

While the use of an AOS provides many benefits, it is important to address its computational cost. The largest penalty comes from the AOS trying to identify the most effective operators. Prior to the search, it is unintuitive to estimate the performance of an operator, so the AOS must estimate an operator's performances by applying it, evaluating the offspring solution, and checking the quality of the offspring solution. Because an operator's effect on the search can only be measured by evaluating solutions, an AOS must expend valuable function evaluations to identify the effective operators. Moreover, since the performance of an operator is largely dependent on the state of the search, the AOS must continually monitor the operators' performances, which dedicates more function evaluations for identifying the effective operators instead of for conducting the search on the problem at hand. This apparent flaw is observed in the minimum selection probability of operator selection strategies that forces the AOS to occasionally apply poorly performing operators in case the state of the search has changed and they begin to perform well.

The cost of identifying the most effective operators has measurable consequences on an MOEA's search performance. The results from Chapter 2 showed that on many occasions, an AOS could not outperform an MOEA that utilized the single best operator for a given problem. Given multiple operators, an AOS must continually evaluate the performance of all operators and cannot fully exploit the most effective operator. Prior to the search, however, correctly identifying the single best operator for a given problem and MOEA is difficult or impossible. Therefore, an AOS sacrifices maximal search performance for a more robust and adaptive search strategy that generalizes better over a range of problems.

A major implication of the costs of identifying effective operators is the limitation on the amount of knowledge that can be incorporated into knowledge-intensive MOEAs and KDO algorithms. With the proposed method of using an AOS to incorporate knowledge-dependent operators, the amount of knowledge that can be incorporated depends on the number of knowledge-dependent operators provided to the MOEA. Given many operators, however, the AOS will have a more difficult time identifying the most effective ones. Moreover, since the performance of the operators are subject to change during the optimization, the AOS must utilize the limited function evaluations to continually monitor the many operators' performances. Therefore, a developer or user of a knowledge-intensive MOEA is responsible for using his or her best judgement to select a few of the most promising design heuristics to apply. Similarly, when using an AOS for a KDO algorithm, the number of newly created knowledge-dependent operators should be limited. Unfortunately, there are no studies examining the effect of the number of operators on the performance of an AOS, so the limit on the number of operators used by an AOS is not well understood. The best guidance for selecting the maximum number of operators comes from effective AOS in the literature, many of which use between 2 and 6 operators [68, 73, 82, 109, 113, 125] and one example that uses 20 operators [122].

Finally, the strength of the conclusions drawn in this thesis that argue for the benefits of incorporate knowledge through an AOS are limited by the number of test problems used. In order to replicate a realistic use of a knowledge-intensive MOEA or KDO algorithm, it was necessary to apply the algorithms to a real-world optimization problem as opposed to standard benchmarking problems. Real-world problems are challenging to solve due to the presence of complex interactions between the decision variables and the objective values, and the difficulty of such problems provides a stronger motivation for more efficient search algorithms like knowledge-intensive MOEAs or KDO algorithms. Moreover, a real-world problem is required when incorporating expert knowledge into the algorithm or assessing the comprehensibility of information extracted by a data mining algorithm. Utilizing knowledge-dependent operators that resemble expert knowledge provided a realistic example in Chapter 3 where some knowledge was not beneficial in improving the search performance. Such realism cannot be replicated by using standard benchmarking problems and fabricating operators that capture information on how to improve solutions because these mathematical problems have no domain context. The downside of using a real-world optimization problem with complex interactions is that it takes time to define these problems and gather design heuristics used in the domain, which limited the number of problems that were used in this thesis. The DSM design problems, however, were well-developed and of interest to the community, and the algorithms could be supplied with a rich set of knowledge for the experiments.

## 5.4 Future work

There are many opportunities for future work in the area of AOS strategies, knowledge-intensive MOEAs, and KDO algorithms due to a number of factors. First, the use of AOS strategies has not become ubiquitous in MOEAs despite over a decade of related work demonstrating its benefits. Second, the research community of algorithm developers seem to be disconnected from the researchers who use the optimization algorithms on real-world problems. This makes it difficult to advance knowledge-intensive and KDO algorithms because developers focus on solving standard benchmarking problems that have little resemblance of real-world problems and users do not have the algorithmic background necessary to leverage their expertise for the optimization. More collaboration between the two groups could spur greater improvements to multiobjective optimization. Finally, research and development of KDO algorithms are still scarce, which leaves many opportunities for future work.

One area that requires greater attention is understanding the limits on how many operators an AOS can effectively utilize in a search. A diversity of operators elevates the generality of an MOEA, but including too many operators degrades the ability of the AOS to identify the effective ones. Currently, there are no known studies examining the effect of the number of operators on an AOS's performance. Although there is likely no optimal number of operators for a given AOS strategy, an approximate limit would inform developers on how many knowledge-dependent operators could be utilized within knowledge-intensive MOEAs and KDO algorithms.

A question raised in Chapter 3 was the effect of the quality of the provided knowledge on an algorithm's search performance. The analysis of the experiments revealed some of the knowledge was more applicable to solutions in specific regions of the tradespace while other knowledge was more general and captured the design decisions of solutions from a larger region of the tradespace. Does design knowledge that is specific to a small region of the tradespace only containing high-quality solutions lead to a more effective knowledge-dependent operator? Or does design knowledge describing a large portion of the tradespace with both high-quality and low-quality solutions result in a knowledge-dependent operator that is effective for over a longer period? Knowledge that tries to push solutions to a small and specific region of the tradespace is likely limited in the types of solutions it can generate and would not be helpful for much of the search. If the MOEA has enough explorative capability, however, finding a few high-quality solutions in one region of the tradespace may help identify other high-quality solutions in the neighborhood. In contrast, more general knowledge that captures a larger portion of the tradespace including both high-quality and low-quality solutions will be applicable for a longer duration of the search. Since the knowledge can lead to low-quality solutions, however, it will not always succeed in improving a solution. This tradeoff between specificity and generality relates to the support and confidence metrics discussed in Chapter 4. A better understanding of how these qualities affect an algorithm's performance would allow for KDO algorithms to better select which of the extracted information to apply for the search.

KDO algorithms might also benefit from learning how to map knowledge-dependent operators to solutions from specific regions of the tradespace or solutions that exhibit a specific design feature. For example, the analysis in Chapter 3 showed that some of the knowledge captured only a portion of the Pareto front. In current AOS strategies a high-performing operator will be applied to the selected parent solutions regardless of any of the parent solutions' properties or traits. This means that knowledge that is only useful for a small set of solutions can get applied to any solution, which is not an efficient use of that knowledge-dependent operator. Instead, if an AOS could select its knowledge-dependent operators with some consideration to the selected parent solution's design features, then the knowledge could be used more efficiently. During the data mining phases, the KDO algorithm could approximate the mapping of the knowledge-dependent operators to certain areas of the tradespace, it could more efficiently utilize the extracted knowledge.

Finally, the methods developed and presented in this thesis should be applied to more real-world problems to generalize the findings. This thesis examined two DSM design problems that were closely related. Other problems will present new insights and challenges that will likely require variations on the presented methods. If these algorithm variants can also show the benefits of using an AOS to utilize knowledge-dependent operators, it paves a brighter future for further advancements to combine optimization algorithms with human expertise.

APPENDIX A

# APPENDIX A

Table A.1: Experiment B: mean (standard deviation) IGD values achieved by decomposition-based AOS

| Problem | Default | Best | Random | PM-OP-De | PM-SI-De | PM-CS-De | AP-OP-De | AP-SI-De | AP-CS-De |
|---|---|---|---|---|---|---|---|---|---|
| WFG1 | 0.4651 (0.007) | 0.4151† (0.023) | 0.3897† (0.005) | 0.3891† (0.008) | 0.3404† (0.023) | 0.3431† (0.018) | 0.3919† (0.010) | **0.3323**† (0.019) | 0.3386† (0.012) |
| WFG2 | 0.0126 (0.001) | 0.0106† (0.000) | 0.0107† (0.000) | 0.0107† (0.000) | 0.0107† (0.001) | 0.0105† (0.000) | 0.0107† (0.000) | 0.0106† (0.000) | **0.0105**† (0.000) |
| WFG3 | 0.0343 (0.000) | **0.0338**† (0.000) | 0.0342 (0.000) | 0.0342 (0.000) | 0.0341† (0.000) | 0.0341† (0.000) | 0.0343 (0.000) | 0.0342 (0.000) | 0.0342 (0.000) |
| WFG4 | 0.0202 (0.002) | **0.0060**† (0.000) | 0.0075† (0.001) | 0.0073† (0.000) | 0.0070† (0.001) | 0.0064† (0.000) | 0.0073† (0.001) | 0.0067† (0.000) | 0.0063† (0.000) |
| WFG5 | 0.0277 (0.000) | 0.0277 (0.000) | 0.0278 (0.000) | 0.0278 (0.000) | 0.0279 (0.000) | 0.0277 (0.000) | 0.0278 (0.000) | 0.0277 (0.000) | **0.0275**† (0.001) |
| WFG6 | 0.0219 (0.029) | 0.0201 (0.005) | 0.0185 (0.006) | **0.0162** (0.005) | 0.0188 (0.005) | 0.0174 (0.005) | 0.0171 (0.006) | 0.0186 (0.006) | 0.0184 (0.006) |
| WFG7 | 0.0057 (0.000) | **0.0052**† (0.000) | 0.0054† (0.000) | 0.0054† (0.000) | 0.0054† (0.000) | 0.0053† (0.000) | 0.0055† (0.000) | 0.0053† (0.000) | 0.0053† (0.000) |
| WFG8 | **0.0349** (0.003) | 0.0349 (0.003) | 0.0396 (0.001) | 0.0393 (0.001) | 0.0395 (0.001) | 0.0394 (0.001) | 0.0396 (0.001) | 0.0392 (0.001) | 0.0401 (0.001) |
| WFG9 | 0.0108 (0.001) | **0.0092**† (0.002) | 0.0100† (0.002) | 0.0096† (0.001) | 0.0100† (0.003) | 0.0097† (0.002) | 0.0096† (0.001) | 0.0092† (0.001) | 0.0097† (0.001) |
| UF1 | 0.0029 (0.002) | 0.0029 (0.002) | 0.0045 (0.003) | 0.0044 (0.004) | **0.0027** (0.002) | 0.0050 (0.004) | 0.0031 (0.002) | 0.0030 (0.002) | 0.0059 (0.005) |
| UF2 | **0.0113** (0.005) | 0.0113 (0.005) | 0.0146 (0.006) | 0.0142 (0.004) | 0.0142 (0.007) | 0.0141 (0.005) | 0.0148 (0.006) | 0.0133 (0.006) | 0.0151 (0.003) |
| UF3 | **0.0338** (0.029) | 0.0338 (0.029) | 0.0621 (0.014) | 0.0554 (0.015) | 0.0465 (0.015) | 0.0699 (0.021) | 0.0582 (0.019) | 0.0487 (0.015) | 0.0708 (0.021) |
| UF4 | 0.0566 (0.004) | 0.0359† (0.003) | 0.0351† (0.002) | 0.0353† (0.002) | **0.0340**† (0.002) | 0.0348† (0.002) | 0.0361† (0.002) | 0.0346† (0.003) | 0.0352† (0.002) |
| UF5 | 0.3122 (0.039) | 0.2079† (0.031) | 0.1918† (0.041) | **0.1890**† (0.044) | 0.1989† (0.047) | 0.1926† (0.050) | 0.1937† (0.053) | 0.1915† (0.053) | 0.1927† (0.041) |
| UF6 | 0.1332 (0.093) | 0.1151 (0.035) | 0.1185 (0.080) | 0.1098 (0.027) | 0.1106 (0.033) | **0.1080** (0.036) | 0.1101 (0.069) | 0.1303 (0.074) | 0.1154 (0.061) |
| UF7 | 0.0080 (0.003) | 0.0080 (0.003) | 0.0065† (0.004) | 0.0064 (0.004) | **0.0047**† (0.002) | 0.0068 (0.003) | 0.0068 (0.003) | 0.0060† (0.004) | 0.0065 (0.003) |
| UF8 | **0.0550** (0.009) | 0.0550 (0.009) | 0.1107 (0.028) | 0.1089 (0.023) | 0.1177 (0.037) | 0.1073 (0.028) | 0.1071 (0.019) | 0.1119 (0.030) | 0.1189 (0.047) |
| UF9 | 0.0582 (0.044) | **0.0520** (0.021) | 0.0904 (0.050) | 0.0828 (0.045) | 0.0773 (0.043) | 0.0806 (0.045) | 0.0910 (0.050) | 0.0919 (0.049) | 0.0999 (0.053) |
| UF10 | 0.5053 (0.051) | **0.2010**† (0.046) | 0.2528† (0.091) | 0.2606† (0.096) | 0.2527† (0.090) | 0.2270† (0.076) | 0.2619† (0.101) | 0.2898† (0.110) | 0.2757† (0.106) |
| DTLZ1 | 0.0889 (0.110) | 0.0603† (0.001) | 0.0711† (0.059) | 0.0789† (0.099) | 0.0602† (0.000) | **0.0602**† (0.001) | 0.0604† (0.001) | 0.0604† (0.001) | 0.0602† (0.001) |
| DTLZ2 | 0.0683 (0.001) | 0.0674† (0.000) | 0.0674† (0.001) | 0.0672† (0.001) | 0.0672† (0.001) | **0.0670**† (0.000) | 0.0670† (0.001) | 0.0671† (0.000) | 0.0670† (0.001) |
| DTLZ3 | 4.5023 (8.736) | **0.3013** (0.446) | 2.4008 (1.720) | 2.3044 (1.940) | 0.6498 (0.712) | 1.2224 (1.091) | 2.5593 (1.618) | 0.4560 (0.581) | 0.6291 (0.660) |
| DTLZ4 | **0.0546** (0.005) | 0.0546 (0.005) | 0.0719 (0.004) | 0.0710 (0.004) | 0.0713 (0.005) | 0.0741 (0.004) | 0.0698 (0.005) | 0.0716 (0.004) | 0.0735 (0.004) |
| DTLZ5 | **0.0155** (0.000) | 0.0155 (0.000) | 0.0158 (0.000) | 0.0158 (0.000) | 0.0158 (0.000) | 0.0159 (0.000) | 0.0158 (0.000) | 0.0159 (0.000) | 0.0159 (0.000) |
| DTLZ6 | **0.0149** (0.000) | 0.0149 (0.000) | 0.0151 (0.000) | 0.0151 (0.000) | 0.0151 (0.000) | 0.0151 (0.000) | 0.0151 (0.000) | 0.0151 (0.000) | 0.0151 (0.000) |
| DTLZ7 | 0.1306 (0.037) | 0.1192 (0.001) | 0.1187 (0.001) | **0.1180**† (0.001) | 0.1190 (0.001) | 0.1186 (0.001) | 0.1185 (0.002) | 0.1191 (0.001) | 0.1184 (0.001) |

Using a Wilcoxon rank sum test with 0.05 significance level, † indicates significantly better IGD than MOEA/D-DRA with its default operator (DE).

Boldface entries indicate best mean value.

Table A.2: Experiment B: mean (standard deviation) HV values achieved by decomposition-based AOS

| Problem | Default | Best | Random | PM-OP-De | PM-SI-De | PM-CS-De | AP-OP-De | AP-SI-De | AP-CS-De |
|---|---|---|---|---|---|---|---|---|---|
| WFG1 | 0.2475 (0.005) | 0.2992$^{\dagger}$ (0.017) | 0.3186$^{\dagger}$ (0.006) | 0.3183$^{\dagger}$ (0.007) | 0.3690$^{\dagger}$ (0.025) | 0.3663$^{\dagger}$ (0.018) | 0.3153$^{\dagger}$ (0.010) | **0.3775**$^{\dagger}$ (0.020) | 0.3719$^{\dagger}$ (0.011) |
| WFG2 | 0.7681 (0.001) | 0.7725$^{\dagger}$ (0.001) | 0.7717$^{\dagger}$ (0.001) | 0.7719$^{\dagger}$ (0.001) | 0.7720$^{\dagger}$ (0.002) | 0.7725$^{\dagger}$ (0.001) | 0.7719$^{\dagger}$ (0.001) | 0.7724$^{\dagger}$ (0.001) | **0.7726**$^{\dagger}$ (0.001) |
| WFG3 | 0.6161 (0.000) | **0.6171**$^{\dagger}$ (0.000) | 0.6162 (0.000) | 0.6161 (0.001) | 0.6164$^{\dagger}$ (0.000) | 0.6164$^{\dagger}$ (0.000) | 0.6159 (0.001) | 0.6163 (0.000) | 0.6162 (0.001) |
| WFG4 | 0.4046 (0.003) | **0.4310**$^{\dagger}$ (0.000) | 0.4264$^{\dagger}$ (0.001) | 0.4266$^{\dagger}$ (0.001) | 0.4278$^{\dagger}$ (0.001) | 0.4292$^{\dagger}$ (0.001) | 0.4267$^{\dagger}$ (0.001) | 0.4286$^{\dagger}$ (0.001) | 0.4296$^{\dagger}$ (0.001) |
| WFG5 | 0.3976 (0.001) | 0.4006 (0.003) | 0.4003$^{\dagger}$ (0.003) | 0.4008$^{\dagger}$ (0.003) | **0.4016**$^{\dagger}$ (0.003) | 0.4013$^{\dagger}$ (0.003) | 0.4007$^{\dagger}$ (0.003) | 0.4010$^{\dagger}$ (0.003) | 0.4008$^{\dagger}$ (0.003) |
| WFG6 | 0.3878 (0.047) | 0.3886 (0.008) | 0.3915 (0.010) | **0.3955** (0.009) | 0.3910 (0.008) | 0.3935 (0.010) | 0.3941 (0.011) | 0.3915 (0.011) | 0.3919 (0.011) |
| WFG7 | 0.4170 (0.000) | **0.4190**$^{\dagger}$ (0.000) | 0.4181$^{\dagger}$ (0.000) | 0.4181$^{\dagger}$ (0.000) | 0.4182$^{\dagger}$ (0.000) | 0.4185$^{\dagger}$ (0.000) | 0.4179$^{\dagger}$ (0.000) | 0.4183$^{\dagger}$ (0.000) | 0.4185$^{\dagger}$ (0.000) |
| WFG8 | **0.3497** (0.003) | 0.3497 (0.003) | 0.3444 (0.001) | 0.3449 (0.001) | 0.3446 (0.002) | 0.3449 (0.001) | 0.3444 (0.002) | 0.3450 (0.001) | 0.3437 (0.002) |
| WFG9 | 0.4452 (0.003) | 0.4510$^{\dagger}$ (0.005) | 0.4495$^{\dagger}$ (0.005) | 0.4503$^{\dagger}$ (0.004) | 0.4509$^{\dagger}$ (0.006) | 0.4504$^{\dagger}$ (0.005) | 0.4519$^{\dagger}$ (0.004) | **0.4521**$^{\dagger}$ (0.004) | 0.4516$^{\dagger}$ (0.005) |
| UF1 | **0.8716** (0.003) | 0.8716 (0.003) | 0.8679 (0.008) | 0.8679 (0.008) | 0.8713 (0.003) | 0.8670 (0.008) | 0.8696 (0.005) | 0.8712 (0.003) | 0.8638 (0.015) |
| UF2 | **0.8592** (0.007) | 0.8592 (0.007) | 0.8553 (0.007) | 0.8552 (0.005) | 0.8562 (0.007) | 0.8550 (0.006) | 0.8553 (0.006) | 0.8568 (0.006) | 0.8538 (0.005) |
| UF3 | **0.8287** (0.036) | 0.8287 (0.036) | 0.7554 (0.040) | 0.7656 (0.032) | 0.7752 (0.035) | 0.7336 (0.044) | 0.7535 (0.046) | 0.7722 (0.034) | 0.7209 (0.042) |
| UF4 | 0.4467 (0.006) | 0.4746$^{\dagger}$ (0.005) | 0.4846$^{\dagger}$ (0.001) | 0.4846$^{\dagger}$ (0.002) | **0.4863**$^{\dagger}$ (0.002) | 0.4853$^{\dagger}$ (0.001) | 0.4840$^{\dagger}$ (0.002) | 0.4858$^{\dagger}$ (0.002) | 0.4836$^{\dagger}$ (0.003) |
| UF5 | 0.1463 (0.068) | 0.3598$^{\dagger}$ (0.052) | 0.3699$^{\dagger}$ (0.070) | **0.3883**$^{\dagger}$ (0.051) | 0.3756$^{\dagger}$ (0.073) | 0.3825$^{\dagger}$ (0.069) | 0.3735$^{\dagger}$ (0.083) | 0.3866$^{\dagger}$ (0.068) | 0.3868$^{\dagger}$ (0.055) |
| UF6 | 0.4025 (0.093) | 0.4306 (0.046) | 0.4354 (0.051) | 0.4455 (0.039) | 0.4325 (0.058) | 0.4377 (0.042) | **0.4593**$^{\dagger}$ (0.042) | 0.4225 (0.067) | 0.4381 (0.044) |
| UF7 | 0.6944 (0.007) | 0.6944 (0.007) | 0.6971 (0.006) | 0.6973 (0.007) | **0.7004**$^{\dagger}$ (0.004) | 0.6963 (0.005) | 0.6969 (0.006) | 0.6982$^{\dagger}$ (0.006) | 0.6972 (0.006) |
| UF8 | **0.6881** (0.019) | 0.6881 (0.019) | 0.6409 (0.049) | 0.6488 (0.037) | 0.6323 (0.062) | 0.6370 (0.054) | 0.6453 (0.037) | 0.6419 (0.050) | 0.6193 (0.080) |
| UF9 | **1.0072** (0.067) | 1.0072 (0.067) | 0.9435 (0.070) | 0.9537 (0.064) | 0.9589 (0.061) | 0.9562 (0.061) | 0.9399 (0.071) | 0.9418 (0.069) | 0.9301 (0.077) |
| UF10 | 0.0885 (0.022) | 0.3823$^{\dagger}$ (0.096) | 0.3486$^{\dagger}$ (0.096) | 0.3405$^{\dagger}$ (0.096) | 0.3495$^{\dagger}$ (0.097) | **0.3825**$^{\dagger}$ (0.090) | 0.3418$^{\dagger}$ (0.110) | 0.3246$^{\dagger}$ (0.103) | 0.3337$^{\dagger}$ (0.108) |
| DTLZ1 | 1.0206 (0.175) | **1.0840**$^{\dagger}$ (0.004) | 1.0650$^{\dagger}$ (0.062) | 1.0455$^{\dagger}$ (0.156) | 1.0784$^{\dagger}$ (0.001) | 1.0779$^{\dagger}$ (0.003) | 1.0743$^{\dagger}$ (0.006) | 1.0781$^{\dagger}$ (0.003) | 1.0802$^{\dagger}$ (0.001) |
| DTLZ2 | 0.7081 (0.004) | **0.7127**$^{\dagger}$ (0.002) | 0.7105$^{\dagger}$ (0.002) | 0.7113$^{\dagger}$ (0.002) | 0.7105$^{\dagger}$ (0.002) | 0.7110$^{\dagger}$ (0.002) | 0.7108$^{\dagger}$ (0.002) | 0.7105$^{\dagger}$ (0.002) | 0.7110$^{\dagger}$ (0.002) |
| DTLZ3 | 0.3055 (0.299) | **0.5104** (0.234) | 0.0477 (0.137) | 0.0888 (0.200) | 0.3755 (0.335) | 0.1796 (0.266) | 0.0419 (0.148) | 0.4329 (0.300) | 0.3463 (0.329) |
| DTLZ4 | 0.6862 (0.040) | 0.6862 (0.040) | 0.7155$^{\dagger}$ (0.002) | 0.7169$^{\dagger}$ (0.003) | 0.7167$^{\dagger}$ (0.002) | 0.7135 (0.003) | **0.7172**$^{\dagger}$ (0.003) | 0.7169$^{\dagger}$ (0.003) | 0.7149$^{\dagger}$ (0.003) |
| DTLZ5 | 0.2610 (0.000) | 0.2621$^{\dagger}$ (0.000) | 0.2620$^{\dagger}$ (0.000) | 0.2620$^{\dagger}$ (0.000) | 0.2620$^{\dagger}$ (0.000) | 0.2621$^{\dagger}$ (0.000) | 0.2620$^{\dagger}$ (0.000) | 0.2621$^{\dagger}$ (0.000) | **0.2621**$^{\dagger}$ (0.000) |
| DTLZ6 | **0.2643** (0.000) | 0.2643 (0.000) | 0.2642 (0.000) | 0.2642 (0.000) | 0.2642 (0.000) | 0.2642 (0.000) | 0.2642 (0.000) | 0.2642 (0.000) | 0.2642 (0.000) |
| DTLZ7 | 0.4015 (0.032) | **0.4661**$^{\dagger}$ (0.000) | 0.4652$^{\dagger}$ (0.001) | 0.4655$^{\dagger}$ (0.001) | 0.4655$^{\dagger}$ (0.001) | 0.4655$^{\dagger}$ (0.001) | 0.4651$^{\dagger}$ (0.001) | 0.4653$^{\dagger}$ (0.001) | 0.4657$^{\dagger}$ (0.001) |

Using a Wilcoxon rank sum test with 0.05 significance level, † indicates significantly better IGD than MOEA/D-DRA with its default operator (DE).

Boldface entries indicates best mean value.

Table A.3: Experiment B: mean (standard deviation) IGD achieved by dominance-based AOS

| Problem | Default | Best | Random | PM-OP-Do | PM-SI-Do | PM-CS-Do | AP-OP-Do | AP-SI-Do | AP-CS-Do |
|---|---|---|---|---|---|---|---|---|---|
| WFG1 | 0.4195 (0.023) | 0.4195 (0.023) | 0.4683 (0.009) | 0.4097 (0.009) | 0.4097 (0.009) | 0.4168 (0.010) | **0.4054**$^{\dagger}$ (0.013) | 0.4054 (0.013) | 0.4084 (0.011) |
| WFG2 | 0.0037 (0.000) | 0.0037 (0.000) | 0.0039 (0.000) | 0.0035$^{\dagger}$ (0.000) | 0.0035 (0.000) | 0.0037 (0.000) | **0.0035**$^{\dagger}$ (0.000) | 0.0035 (0.000) | 0.0037 (0.000) |
| WFG3 | **0.0344** (0.000) | 0.0344 (0.000) | 0.0350 (0.000) | 0.0346 (0.000) | 0.0346 (0.000) | 0.0348 (0.000) | 0.0346 (0.000) | 0.0346 (0.000) | 0.0350 (0.000) |
| WFG4 | **0.0052** (0.000) | 0.0052 (0.000) | 0.0072 (0.000) | 0.0059 (0.000) | 0.0059 (0.000) | 0.0060 (0.000) | 0.0058 (0.000) | 0.0058 (0.000) | 0.0059 (0.000) |
| WFG5 | **0.0277** (0.000) | 0.0277 (0.000) | 0.0287 (0.000) | 0.0282 (0.000) | 0.0282 (0.000) | 0.0285 (0.000) | 0.0282 (0.000) | 0.0282 (0.000) | 0.0285 (0.000) |
| WFG6 | 0.0324 (0.009) | **0.0072**$^{\dagger}$ (0.001) | 0.0131$^{\dagger}$ (0.003) | 0.0154$^{\dagger}$ (0.003) | 0.0154$^{\dagger}$ (0.003) | 0.0146$^{\dagger}$ (0.003) | 0.0138$^{\dagger}$ (0.003) | 0.0138$^{\dagger}$ (0.003) | 0.0135$^{\dagger}$ (0.005) |
| WFG7 | **0.0058** (0.000) | 0.0058 (0.000) | 0.0063 (0.000) | 0.0062 (0.000) | 0.0062 (0.000) | 0.0062 (0.000) | 0.0062 (0.000) | 0.0062 (0.000) | 0.0061 (0.000) |
| WFG8 | 0.0375 (0.001) | **0.0320**$^{\dagger}$ (0.001) | 0.0395 (0.001) | 0.0394 (0.001) | 0.0394 (0.001) | 0.0394 (0.001) | 0.0393 (0.001) | 0.0393 (0.001) | 0.0399 (0.001) |
| WFG9 | **0.0088** (0.002) | 0.0088 (0.002) | 0.0108 (0.001) | 0.0103 (0.001) | 0.0103 (0.001) | 0.0100 (0.001) | 0.0102 (0.001) | 0.0102 (0.001) | 0.0101 (0.001) |
| UF1 | 0.0738 (0.014) | 0.0306$^{\dagger}$ (0.006) | 0.0368$^{\dagger}$ (0.012) | 0.0276$^{\dagger}$ (0.013) | 0.0276$^{\dagger}$ (0.013) | 0.0310$^{\dagger}$ (0.015) | **0.0276**$^{\dagger}$ (0.013) | 0.0276$^{\dagger}$ (0.013) | 0.0315$^{\dagger}$ (0.013) |
| UF2 | 0.0221 (0.004) | 0.0179$^{\dagger}$ (0.003) | 0.0111$^{\dagger}$ (0.002) | 0.0102$^{\dagger}$ (0.001) | 0.0102$^{\dagger}$ (0.001) | 0.0099$^{\dagger}$ (0.001) | 0.0104$^{\dagger}$ (0.001) | 0.0104$^{\dagger}$ (0.001) | **0.0096**$^{\dagger}$ (0.001) |
| UF3 | 0.0977 (0.019) | 0.0511$^{\dagger}$ (0.017) | 0.0609$^{\dagger}$ (0.025) | 0.0731$^{\dagger}$ (0.020) | 0.0731$^{\dagger}$ (0.020) | 0.0478$^{\dagger}$ (0.018) | 0.0573$^{\dagger}$ (0.023) | 0.0573$^{\dagger}$ (0.023) | **0.0443**$^{\dagger}$ (0.020) |
| UF4 | 0.0419 (0.000) | **0.0306**$^{\dagger}$ (0.001) | 0.0385$^{\dagger}$ (0.000) | 0.0384$^{\dagger}$ (0.000) | 0.0384$^{\dagger}$ (0.000) | 0.0377$^{\dagger}$ (0.000) | 0.0378$^{\dagger}$ (0.000) | 0.0378$^{\dagger}$ (0.000) | 0.0386$^{\dagger}$ (0.000) |
| UF5 | 0.2113 (0.042) | 0.1742$^{\dagger}$ (0.016) | 0.1676$^{\dagger}$ (0.007) | 0.1673$^{\dagger}$ (0.012) | 0.1673$^{\dagger}$ (0.012) | 0.1685$^{\dagger}$ (0.015) | 0.1679$^{\dagger}$ (0.028) | 0.1679$^{\dagger}$ (0.028) | **0.1639**$^{\dagger}$ (0.017) |
| UF6 | 0.1024 (0.016) | 0.1024 (0.016) | 0.1026 (0.008) | 0.1056 (0.016) | 0.1056 (0.016) | 0.1049 (0.021) | **0.1005** (0.011) | 0.1005 (0.011) | 0.1142 (0.036) |
| UF7 | 0.0344 (0.007) | 0.0125$^{\dagger}$ (0.001) | 0.0140$^{\dagger}$ (0.002) | 0.0144$^{\dagger}$ (0.001) | 0.0144$^{\dagger}$ (0.001) | 0.0121$^{\dagger}$ (0.002) | 0.0155$^{\dagger}$ (0.002) | 0.0155$^{\dagger}$ (0.002) | **0.0119**$^{\dagger}$ (0.002) |
| UF8 | 0.1765 (0.018) | **0.1050**$^{\dagger}$ (0.007) | 0.1170$^{\dagger}$ (0.045) | 0.1309$^{\dagger}$ (0.052) | 0.1309$^{\dagger}$ (0.052) | 0.1371 (0.053) | 0.1403 (0.052) | 0.1403$^{\dagger}$ (0.052) | 0.1128$^{\dagger}$ (0.045) |
| UF9 | 0.1290 (0.041) | **0.0879**$^{\dagger}$ (0.012) | 0.1939 (0.068) | 0.1850 (0.073) | 0.1850 (0.073) | 0.2352 (0.077) | 0.1597 (0.063) | 0.1597 (0.063) | 0.2264 (0.073) |
| UF10 | 0.2436 (0.027) | 0.2436 (0.027) | **0.2370** (0.017) | 0.2383 (0.018) | 0.2383 (0.018) | 0.2694 (0.020) | 0.2496 (0.015) | 0.2496 (0.015) | 0.3102 (0.099) |
| DTLZ1 | **0.0929** (0.133) | 0.0929 (0.133) | 0.2403 (0.337) | 0.1248$^{\dagger}$ (0.216) | 0.1248$^{\dagger}$ (0.216) | 0.1245$^{\dagger}$ (0.189) | 0.1381$^{\dagger}$ (0.192) | 0.1381 (0.192) | 0.1498$^{\dagger}$ (0.210) |
| DTLZ2 | 0.0661 (0.003) | **0.0630**$^{\dagger}$ (0.002) | 0.0670 (0.003) | 0.0666 (0.002) | 0.0666 (0.002) | 0.0677 (0.003) | 0.0663 (0.003) | 0.0663 (0.003) | 0.0668 (0.003) |
| DTLZ3 | 4.8399 (2.199) | 4.8399 (2.199) | 9.8644 (4.792) | 4.3780 (2.538) | 4.3780 (2.538) | 3.1796$^{\dagger}$ (1.562) | 3.5829$^{\dagger}$ (2.404) | 3.5829$^{\dagger}$ (2.404) | **2.5565**$^{\dagger}$ (1.247) |
| DTLZ4 | 0.0627 (0.006) | **0.0416**$^{\dagger}$ (0.002) | 0.0665 (0.005) | 0.0646 (0.007) | 0.0646 (0.007) | 0.0656 (0.005) | 0.0639 (0.005) | 0.0639 (0.005) | 0.0641 (0.005) |
| DTLZ5 | 0.0062 (0.000) | **0.0060**$^{\dagger}$ (0.000) | 0.0066 (0.000) | 0.0066 (0.000) | 0.0066 (0.000) | 0.0065 (0.000) | 0.0066 (0.000) | 0.0066 (0.000) | 0.0063 (0.000) |
| DTLZ6 | 0.9001 (0.079) | **0.0058**$^{\dagger}$ (0.000) | 0.0061$^{\dagger}$ (0.000) | 0.0061$^{\dagger}$ (0.000) | 0.0061$^{\dagger}$ (0.000) | 0.0060$^{\dagger}$ (0.000) | 0.0061$^{\dagger}$ (0.000) | 0.0061$^{\dagger}$ (0.000) | 0.0060$^{\dagger}$ (0.000) |
| DTLZ7 | **0.0492** (0.002) | 0.0492 (0.002) | 0.0628 (0.004) | 0.0544 (0.003) | 0.0544 (0.003) | 0.0599 (0.004) | 0.0536 (0.003) | 0.0536 (0.003) | 0.0573 (0.009) |

Using a Wilcoxon rank sum test with 0.05 significance level, † indicates significantly better IGD than NSGA-II with its default operator (SBX).

Boldface entries indicates best mean value.

Table A.4: Experiment B: mean (standard deviation) HV achieved by dominance-based AOS

| Problem | Default | Best | Random | PM-OP-Do | PM-SI-Do | PM-CS-Do | AP-OP-Do | AP-SI-Do | AP-CS-Do |
|---|---|---|---|---|---|---|---|---|---|
| WFG1 | **0.3133** (0.014) | 0.3133 (0.014) | 0.2306 (0.013) | 0.2966 (0.010) | 0.2966 (0.010) | 0.2879 (0.014) | 0.3019 (0.015) | 0.3019 (0.015) | 0.2989 (0.013) |
| WFG2 | 0.7741 (0.001) | 0.7741 (0.001) | 0.7735 (0.001) | 0.7746$^\dagger$ (0.000) | 0.7746 (0.000) | 0.7742 (0.001) | **0.7746**$^\dagger$ (0.001) | 0.7746 (0.001) | 0.7740 (0.001) |
| WFG3 | **0.6164** (0.001) | 0.6164 (0.001) | 0.6154 (0.000) | 0.6162 (0.000) | 0.6162 (0.000) | 0.6158 (0.000) | 0.6162 (0.000) | 0.6162 (0.000) | 0.6154 (0.001) |
| WFG4 | **0.4304** (0.001) | 0.4304 (0.001) | 0.4252 (0.001) | 0.4282 (0.001) | 0.4282 (0.001) | 0.4282 (0.001) | 0.4286 (0.001) | 0.4286 (0.001) | 0.4286 (0.001) |
| WFG5 | **0.4042** (0.001) | 0.4042 (0.001) | 0.4015 (0.002) | 0.4025 (0.002) | 0.4025 (0.002) | 0.4020 (0.002) | 0.4029 (0.002) | 0.4029 (0.002) | 0.4026 (0.002) |
| WFG6 | 0.3686 (0.015) | **0.4129**$^\dagger$ (0.001) | 0.4014$^\dagger$ (0.006) | 0.3971$^\dagger$ (0.005) | 0.3971$^\dagger$ (0.005) | 0.3986$^\dagger$ (0.006) | 0.4000$^\dagger$ (0.005) | 0.4000$^\dagger$ (0.005) | 0.4008$^\dagger$ (0.008) |
| WFG7 | **0.4173** (0.000) | 0.4173 (0.000) | 0.4158 (0.000) | 0.4160 (0.000) | 0.4160 (0.000) | 0.4160 (0.001) | 0.4162 (0.000) | 0.4162 (0.000) | 0.4159 (0.000) |
| WFG8 | 0.3469 (0.001) | **0.3513**$^\dagger$ (0.001) | 0.3440 (0.001) | 0.3445 (0.001) | 0.3445 (0.001) | 0.3444 (0.001) | 0.3447 (0.001) | 0.3447 (0.001) | 0.3434 (0.002) |
| WFG9 | **0.4560** (0.003) | 0.4560 (0.003) | 0.4503 (0.003) | 0.4516 (0.002) | 0.4516 (0.002) | 0.4526 (0.002) | 0.4529 (0.002) | 0.4529 (0.002) | 0.4513 (0.003) |
| UF1 | 0.7634 (0.018) | 0.8246$^\dagger$ (0.012) | 0.8161$^\dagger$ (0.019) | **0.8314**$^\dagger$ (0.022) | 0.8314$^\dagger$ (0.022) | 0.8257$^\dagger$ (0.024) | 0.8311$^\dagger$ (0.022) | 0.8311$^\dagger$ (0.022) | 0.8250$^\dagger$ (0.021) |
| UF2 | 0.8431 (0.005) | 0.8504$^\dagger$ (0.003) | 0.8604$^\dagger$ (0.002) | 0.8612$^\dagger$ (0.001) | 0.8612$^\dagger$ (0.001) | 0.8622$^\dagger$ (0.002) | 0.8614$^\dagger$ (0.002) | **0.8614**$^\dagger$ (0.002) | 0.8626$^\dagger$ (0.002) |
| UF3 | 0.6919 (0.026) | 0.7846$^\dagger$ (0.028) | 0.7790$^\dagger$ (0.035) | 0.7611$^\dagger$ (0.028) | 0.7611$^\dagger$ (0.028) | 0.7961$^\dagger$ (0.030) | 0.7862$^\dagger$ (0.032) | 0.7862$^\dagger$ (0.032) | **0.8045**$^\dagger$ (0.029) |
| UF4 | 0.4747 (0.001) | **0.4867**$^\dagger$ (0.001) | 0.4802$^\dagger$ (0.001) | 0.4802$^\dagger$ (0.000) | 0.4802$^\dagger$ (0.000) | 0.4811$^\dagger$ (0.001) | 0.4810$^\dagger$ (0.001) | 0.4810$^\dagger$ (0.001) | 0.4799$^\dagger$ (0.001) |
| UF5 | 0.3332 (0.081) | 0.3966$^\dagger$ (0.025) | 0.3893$^\dagger$ (0.012) | 0.3971$^\dagger$ (0.028) | 0.3971$^\dagger$ (0.028) | 0.3927$^\dagger$ (0.027) | 0.3924$^\dagger$ (0.054) | 0.3924$^\dagger$ (0.054) | **0.4075**$^\dagger$ (0.031) |
| UF6 | **0.4485** (0.030) | 0.4485 (0.030) | 0.4363 (0.025) | 0.4430 (0.023) | 0.4430 (0.023) | 0.4450 (0.030) | 0.4450 (0.036) | 0.4450 (0.036) | 0.4283 (0.039) |
| UF7 | 0.6519 (0.010) | 0.6873$^\dagger$ (0.002) | 0.6852$^\dagger$ (0.003) | 0.6845$^\dagger$ (0.002) | 0.6845$^\dagger$ (0.002) | 0.6885$^\dagger$ (0.003) | 0.6826$^\dagger$ (0.003) | 0.6826$^\dagger$ (0.003) | **0.6889**$^\dagger$ (0.004) |
| UF8 | 0.4628 (0.033) | **0.5721**$^\dagger$ (0.016) | 0.5701$^\dagger$ (0.088) | 0.5404 (0.103) | 0.5404$^\dagger$ (0.103) | 0.5268 (0.105) | 0.5219 (0.101) | 0.5219$^\dagger$ (0.101) | 0.5710$^\dagger$ (0.086) |
| UF9 | 0.8465 (0.059) | **0.9207**$^\dagger$ (0.029) | 0.7307 (0.137) | 0.7472 (0.141) | 0.7472 (0.141) | 0.6293 (0.165) | 0.8078 (0.111) | 0.8078 (0.111) | 0.6547 (0.140) |
| UF10 | 0.2906 (0.068) | **0.3352**$^\dagger$ (0.056) | 0.3123 (0.033) | 0.3102 (0.035) | 0.3102 (0.035) | 0.2686 (0.034) | 0.3026 (0.024) | 0.3026 (0.024) | 0.2023 (0.077) |
| DTLZ1 | **1.0271** (0.213) | 1.0271 (0.213) | 0.8303 (0.403) | 0.9849 (0.277) | 0.9849 (0.277) | 0.9717 (0.302) | 0.9474 (0.315) | 0.9474 (0.315) | 0.9315$^\dagger$ (0.340) |
| DTLZ2 | **0.7037** (0.005) | 0.7037 (0.005) | 0.6946 (0.005) | 0.7009 (0.006) | 0.7009 (0.006) | 0.6943 (0.007) | 0.7024 (0.007) | 0.7024 (0.007) | 0.6941 (0.010) |
| DTLZ3 | 0.0000 (0.000) | 0.0098 (0.031) | 0.0000 (0.000) | 0.0109 (0.060) | 0.0109 (0.060) | 0.0005 (0.003) | 0.0004 (0.002) | 0.0004 (0.002) | **0.0203** (0.097) |
| DTLZ4 | 0.7096 (0.007) | 0.7096 (0.007) | 0.7096 (0.005) | 0.7124 (0.005) | 0.7124 (0.005) | 0.7089 (0.006) | **0.7126** (0.005) | 0.7126 (0.005) | 0.7098 (0.006) |
| DTLZ5 | **0.2667** (0.000) | 0.2667 (0.000) | 0.2657 (0.000) | 0.2660 (0.000) | 0.2660 (0.000) | 0.2661 (0.000) | 0.2660 (0.000) | 0.2660 (0.000) | 0.2661 (0.000) |
| DTLZ6 | 0.0000 (0.000) | **0.2696**$^\dagger$ (0.000) | 0.2695$^\dagger$ (0.000) | 0.2695$^\dagger$ (0.000) | 0.2695$^\dagger$ (0.000) | 0.2695$^\dagger$ (0.000) | 0.2694$^\dagger$ (0.000) | 0.2694$^\dagger$ (0.000) | 0.2694$^\dagger$ (0.000) |
| DTLZ7 | **0.5383** (0.004) | 0.5383 (0.004) | 0.5039 (0.007) | 0.5281 (0.005) | 0.5281 (0.005) | 0.5147 (0.007) | 0.5283 (0.006) | 0.5283 (0.006) | 0.5218 (0.017) |

Using a Wilcoxon rank sum test with 0.05 significance level, † indicates significantly better IGD than NSGA-II with its default operator (SBX).

Boldface entries indicates best mean value.

Table A.5: Experiment B: mean (standard deviation) IGD achieved by indicator-based AOS

| Problem | Default | Best | Random | PM-OP-I | PM-SI-I | PM-CS-I | AP-OP-I | AP-SI-I | AP-CS-I |
|---|---|---|---|---|---|---|---|---|---|
| WFG1 | 0.4121 (0.021) | 0.4121 (0.021) | 0.3980$^\dagger$ (0.012) | 0.4030$^\dagger$ (0.008) | 0.3562$^\dagger$ (0.012) | 0.3603$^\dagger$ (0.013) | 0.4176 (0.013) | **0.3561**$^\dagger$ (0.015) | 0.3646$^\dagger$ (0.018) |
| WFG2 | **0.0185** (0.001) | 0.0185 (0.001) | 0.0197 (0.001) | 0.0200 (0.001) | 0.0196 (0.001) | 0.0198 (0.001) | 0.0198 (0.001) | 0.0196 (0.001) | 0.0197 (0.001) |
| WFG3 | 0.0352 (0.001) | **0.0347**$^\dagger$ (0.001) | 0.0351 (0.001) | 0.0351 (0.001) | 0.0349 (0.001) | 0.0349 (0.001) | 0.0349$^\dagger$ (0.001) | 0.0351 (0.001) | 0.0349$^\dagger$ (0.001) |
| WFG4 | **0.0102** (0.001) | 0.0102 (0.001) | 0.0121 (0.002) | 0.0119 (0.002) | 0.0114 (0.002) | 0.0108 (0.002) | 0.0111 (0.002) | 0.0116 (0.002) | 0.0105 (0.001) |
| WFG5 | 0.0312 (0.001) | **0.0281**$^\dagger$ (0.000) | 0.0286$^\dagger$ (0.001) | 0.0287$^\dagger$ (0.000) | 0.0287$^\dagger$ (0.001) | 0.0287$^\dagger$ (0.000) | 0.0288$^\dagger$ (0.000) | 0.0288$^\dagger$ (0.000) | 0.0288$^\dagger$ (0.001) |
| WFG6 | 0.0353 (0.008) | **0.0128**$^\dagger$ (0.003) | 0.0183$^\dagger$ (0.003) | 0.0185$^\dagger$ (0.004) | 0.0192$^\dagger$ (0.004) | 0.0183$^\dagger$ (0.004) | 0.0170$^\dagger$ (0.004) | 0.0185$^\dagger$ (0.004) | 0.0187$^\dagger$ (0.004) |
| WFG7 | 0.0083 (0.001) | **0.0055**$^\dagger$ (0.000) | 0.0075$^\dagger$ (0.001) | 0.0067$^\dagger$ (0.001) | 0.0071$^\dagger$ (0.001) | 0.0074$^\dagger$ (0.001) | 0.0070$^\dagger$ (0.001) | 0.0073$^\dagger$ (0.001) | 0.0070$^\dagger$ (0.001) |
| WFG8 | 0.0445 (0.004) | **0.0361**$^\dagger$ (0.004) | 0.0438 (0.002) | 0.0426$^\dagger$ (0.002) | 0.0437 (0.002) | 0.0422$^\dagger$ (0.002) | 0.0426 (0.003) | 0.0431 (0.003) | 0.0425 (0.003) |
| WFG9 | 0.0126 (0.002) | **0.0103**$^\dagger$ (0.001) | 0.0113$^\dagger$ (0.001) | 0.0114 (0.002) | 0.0111$^\dagger$ (0.001) | 0.0113$^\dagger$ (0.001) | 0.0116 (0.002) | 0.0112$^\dagger$ (0.001) | 0.0110$^\dagger$ (0.001) |
| UF1 | 0.1014 (0.010) | **0.0435**$^\dagger$ (0.004) | 0.0579$^\dagger$ (0.013) | 0.0636$^\dagger$ (0.013) | 0.0534$^\dagger$ (0.011) | 0.0499$^\dagger$ (0.009) | 0.0658$^\dagger$ (0.013) | 0.0518$^\dagger$ (0.009) | 0.0486$^\dagger$ (0.009) |
| UF2 | 0.0340 (0.006) | 0.0276$^\dagger$ (0.005) | 0.0261$^\dagger$ (0.004) | **0.0271**$^\dagger$ (0.005) | 0.0269$^\dagger$ (0.006) | 0.0252$^\dagger$ (0.005) | 0.0278$^\dagger$ (0.006) | 0.0255$^\dagger$ (0.004) | 0.0249$^\dagger$ (0.003) |
| UF3 | 0.2189 (0.042) | **0.0290**$^\dagger$ (0.013) | 0.1664$^\dagger$ (0.042) | 0.1718$^\dagger$ (0.036) | 0.1574$^\dagger$ (0.047) | 0.1122$^\dagger$ (0.058) | 0.1729$^\dagger$ (0.041) | 0.1419$^\dagger$ (0.062) | 0.1327$^\dagger$ (0.053) |
| UF4 | 0.0479 (0.003) | **0.0359**$^\dagger$ (0.003) | 0.0407$^\dagger$ (0.001) | 0.0412$^\dagger$ (0.002) | 0.0406$^\dagger$ (0.002) | 0.0408$^\dagger$ (0.001) | 0.0415$^\dagger$ (0.001) | 0.0409$^\dagger$ (0.001) | 0.0408$^\dagger$ (0.002) |
| UF5 | 0.2432 (0.042) | **0.1988**$^\dagger$ (0.027) | 0.3274 (0.033) | 0.2978 (0.058) | 0.2833 (0.063) | 0.2778 (0.065) | 0.3106 (0.052) | 0.3008 (0.053) | 0.2946 (0.066) |
| UF6 | 0.1466 (0.053) | 0.1441 (0.025) | 0.1506 (0.056) | 0.1614 (0.064) | 0.1550 (0.058) | 0.1593 (0.072) | 0.1470 (0.047) | **0.1387** (0.031) | 0.1422 (0.046) |
| UF7 | 0.0561 (0.040) | **0.0210**$^\dagger$ (0.002) | 0.0289$^\dagger$ (0.003) | 0.0299$^\dagger$ (0.003) | 0.0281$^\dagger$ (0.002) | 0.0267$^\dagger$ (0.002) | 0.0312$^\dagger$ (0.003) | 0.0274$^\dagger$ (0.001) | 0.0256$^\dagger$ (0.003) |
| UF8 | **0.4139** (0.041) | 0.4139 (0.041) | 0.4456$^\dagger$ (0.048) | 0.4297 (0.040) | 0.4414 (0.055) | 0.4393 (0.038) | 0.4327 (0.025) | 0.4428 (0.036) | 0.4366 (0.051) |
| UF9 | 0.1970 (0.079) | **0.1025**$^\dagger$ (0.046) | 0.2430 (0.043) | 0.2309 (0.054) | 0.2431 (0.043) | 0.2449 (0.041) | 0.2465 (0.039) | 0.2452 (0.041) | 0.2368 (0.053) |
| UF10 | 0.5780 (0.026) | **0.4111**$^\dagger$ (0.061) | 0.5809 (0.036) | 0.5223 (0.099) | 0.5726 (0.065) | 0.5152 (0.098) | 0.5299 (0.098) | 0.5445 (0.092) | 0.5538 (0.076) |
| DTLZ1 | 0.3252 (0.037) | 0.3252 (0.037) | 0.1971$^\dagger$ (0.036) | 0.1902$^\dagger$ (0.034) | 0.1957$^\dagger$ (0.037) | 0.2031$^\dagger$ (0.044) | 0.1911$^\dagger$ (0.038) | 0.1888$^\dagger$ (0.045) | **0.1834**$^\dagger$ (0.035) |
| DTLZ2 | 0.0866 (0.003) | **0.0820**$^\dagger$ (0.003) | 0.0930 (0.004) | 0.0909 (0.004) | 0.0925 (0.004) | 0.0913 (0.003) | 0.0907 (0.004) | 0.0912 (0.004) | 0.0913 (0.004) |
| DTLZ3 | **0.6469** (0.274) | 0.6469 (0.274) | 4.5019 (2.491) | 4.5374 (3.073) | 2.8921 (1.434) | 2.2379 (1.097) | 3.6364 (2.118) | 2.9910 (1.556) | 1.9913 (0.966) |
| DTLZ4 | 0.0719 (0.055) | **0.0503**$^\dagger$ (0.003) | 0.0515$^\dagger$ (0.002) | 0.0517 (0.003) | 0.0520 (0.002) | 0.0516 (0.003) | 0.0515$^\dagger$ (0.002) | 0.0520 (0.003) | 0.0515$^\dagger$ (0.003) |
| DTLZ5 | 0.0181 (0.002) | **0.0152**$^\dagger$ (0.001) | 0.0179 (0.002) | 0.0177 (0.001) | 0.0181 (0.002) | 0.0173 (0.002) | 0.0174 (0.001) | 0.0177 (0.001) | 0.0178 (0.002) |
| DTLZ6 | 0.1638 (0.036) | **0.0384**$^\dagger$ (0.011) | 0.1532 (0.039) | 0.1391$^\dagger$ (0.050) | 0.1197$^\dagger$ (0.034) | 0.1398$^\dagger$ (0.033) | 0.1439$^\dagger$ (0.076) | 0.1444$^\dagger$ (0.076) | 0.1627 (0.100) |
| DTLZ7 | 0.0999 (0.094) | 0.0999 (0.094) | 0.0711 (0.006) | 0.0690 (0.005) | 0.0694 (0.004) | **0.0685** (0.004) | 0.0713 (0.004) | 0.0697 (0.004) | 0.0697 (0.004) |

Using a Wilcoxon rank sum test with 0.05 significance level, † indicates significantly better IGD than IBEA with its default operator (SBX).

Boldface entries indicates best mean value.

Table A.6: Experiment B: mean (standard deviation) HV achieved by indicator-based AOS

| Problem | Default | Best | Random | PM-OP-I | PM-SI-I | PM-CS-I | AP-OP-I | AP-SI-I | AP-CS-I |
|---|---|---|---|---|---|---|---|---|---|
| WFG1 | 0.3082 (0.014) | 0.3082 (0.014) | 0.3072 (0.010) | 0.3021 (0.006) | 0.3515$^\dagger$ (0.013) | 0.3457$^\dagger$ (0.013) | 0.2893 (0.010) | **0.3517**$^\dagger$ (0.016) | 0.3407$^\dagger$ (0.018) |
| WFG2 | 0.7706 (0.001) | 0.7706 (0.001) | 0.7702 (0.001) | 0.7702 (0.001) | 0.7705 (0.001) | **0.7706** (0.000) | 0.7701 (0.001) | 0.7705 (0.001) | 0.7706 (0.001) |
| WFG3 | 0.6149 (0.001) | 0.6157$^\dagger$ (0.001) | 0.6155$^\dagger$ (0.002) | 0.6155$^\dagger$ (0.001) | 0.6157$^\dagger$ (0.001) | **0.6159**$^\dagger$ (0.001) | 0.6159$^\dagger$ (0.001) | 0.6156$^\dagger$ (0.001) | 0.6159$^\dagger$ (0.001) |
| WFG4 | **0.4246** (0.002) | 0.4246 (0.002) | 0.4194 (0.004) | 0.4199 (0.003) | 0.4218 (0.003) | 0.4231 (0.003) | 0.4212 (0.003) | 0.4216 (0.004) | 0.4234 (0.003) |
| WFG5 | 0.3975 (0.003) | 0.4025$^\dagger$ (0.002) | 0.4032$^\dagger$ (0.002) | 0.4025$^\dagger$ (0.002) | 0.4030$^\dagger$ (0.002) | 0.4031$^\dagger$ (0.001) | 0.4028$^\dagger$ (0.001) | 0.4026$^\dagger$ (0.002) | **0.4032**$^\dagger$ (0.001) |
| WFG6 | 0.3648 (0.014) | **0.4072**$^\dagger$ (0.006) | 0.3959$^\dagger$ (0.006) | 0.3957$^\dagger$ (0.008) | 0.3939$^\dagger$ (0.007) | 0.3962$^\dagger$ (0.009) | 0.3982$^\dagger$ (0.008) | 0.3953$^\dagger$ (0.007) | 0.3949$^\dagger$ (0.007) |
| WFG7 | 0.4130 (0.002) | **0.4154**$^\dagger$ (0.001) | 0.4136 (0.002) | 0.4150$^\dagger$ (0.001) | 0.4143$^\dagger$ (0.001) | 0.4140$^\dagger$ (0.001) | 0.4144$^\dagger$ (0.002) | 0.4141$^\dagger$ (0.002) | 0.4145$^\dagger$ (0.001) |
| WFG8 | 0.3346 (0.007) | **0.3451**$^\dagger$ (0.006) | 0.3367 (0.004) | 0.3390$^\dagger$ (0.004) | 0.3372$^\dagger$ (0.004) | 0.3396$^\dagger$ (0.004) | 0.3388$^\dagger$ (0.005) | 0.3380$^\dagger$ (0.006) | 0.3391$^\dagger$ (0.004) |
| WFG9 | 0.4500 (0.005) | 0.4507 (0.005) | 0.4512 (0.003) | 0.4511 (0.003) | 0.4514 (0.003) | 0.4511 (0.003) | 0.4511 (0.003) | 0.4511 (0.003) | **0.4516** (0.004) |
| UF1 | 0.7325 (0.014) | **0.8145**$^\dagger$ (0.006) | 0.7959$^\dagger$ (0.020) | 0.7873$^\dagger$ (0.020) | 0.8030$^\dagger$ (0.016) | 0.8080$^\dagger$ (0.014) | 0.7835$^\dagger$ (0.018) | 0.8039$^\dagger$ (0.015) | 0.8093$^\dagger$ (0.014) |
| UF2 | 0.8357 (0.007) | 0.8379$^\dagger$ (0.004) | 0.8441$^\dagger$ (0.005) | 0.8426$^\dagger$ (0.005) | 0.8441$^\dagger$ (0.005) | 0.8458$^\dagger$ (0.005) | 0.8424$^\dagger$ (0.006) | 0.8453$^\dagger$ (0.005) | **0.8461**$^\dagger$ (0.004) |
| UF3 | 0.6011 (0.043) | **0.8262**$^\dagger$ (0.022) | 0.5961 (0.083) | 0.5773 (0.058) | 0.6096 (0.092) | 0.6907$^\dagger$ (0.104) | 0.5871 (0.072) | 0.6337 (0.112) | 0.6525$^\dagger$ (0.094) |
| UF4 | 0.4638 (0.004) | **0.4807**$^\dagger$ (0.004) | 0.4759$^\dagger$ (0.002) | 0.4740$^\dagger$ (0.002) | 0.4758$^\dagger$ (0.002) | 0.4758$^\dagger$ (0.002) | 0.4732$^\dagger$ (0.002) | 0.4761$^\dagger$ (0.001) | 0.4759$^\dagger$ (0.002) |
| UF5 | 0.3291 (0.058) | **0.3685**$^\dagger$ (0.041) | 0.2209 (0.043) | 0.2569 (0.075) | 0.2755 (0.082) | 0.2888 (0.092) | 0.2399 (0.064) | 0.2581 (0.075) | 0.2655 (0.087) |
| UF6 | 0.4226 (0.063) | 0.4226 (0.063) | 0.4144 (0.056) | 0.4094 (0.056) | 0.4211 (0.058) | 0.4173 (0.058) | 0.4126 (0.059) | 0.4306 (0.032) | **0.4370** (0.050) |
| UF7 | 0.6300 (0.041) | **0.6787**$^\dagger$ (0.002) | 0.6678$^\dagger$ (0.004) | 0.6660$^\dagger$ (0.005) | 0.6690$^\dagger$ (0.003) | 0.6711$^\dagger$ (0.002) | 0.6641$^\dagger$ (0.005) | 0.6700$^\dagger$ (0.002) | 0.6726$^\dagger$ (0.004) |
| UF8 | 0.4600 (0.014) | **0.4638** (0.003) | 0.4132 (0.055) | 0.4392 (0.051) | 0.4095 (0.068) | 0.4204 (0.057) | 0.4445 (0.025) | 0.4229 (0.057) | 0.4341 (0.059) |
| UF9 | 0.7929 (0.077) | **0.9389**$^\dagger$ (0.045) | 0.8403$^\dagger$ (0.029) | 0.8491$^\dagger$ (0.041) | 0.8398$^\dagger$ (0.034) | 0.8450$^\dagger$ (0.036) | 0.8436$^\dagger$ (0.031) | 0.8435$^\dagger$ (0.036) | 0.8512$^\dagger$ (0.044) |
| UF10 | 0.2323 (0.020) | **0.3059**$^\dagger$ (0.060) | 0.2384$^\dagger$ (0.029) | 0.2792$^\dagger$ (0.074) | 0.2420$^\dagger$ (0.043) | 0.2889$^\dagger$ (0.079) | 0.2789$^\dagger$ (0.077) | 0.2693$^\dagger$ (0.071) | 0.2536$^\dagger$ (0.054) |
| DTLZ1 | 0.6073 (0.086) | 0.6073 (0.086) | 0.8124$^\dagger$ (0.072) | 0.8231$^\dagger$ (0.084) | 0.8410$^\dagger$ (0.071) | 0.8121$^\dagger$ (0.075) | 0.8341$^\dagger$ (0.076) | 0.8357$^\dagger$ (0.091) | **0.8529**$^\dagger$ (0.071) |
| DTLZ2 | 0.7415 (0.004) | 0.7415 (0.004) | 0.7400 (0.003) | 0.7407 (0.003) | 0.7408 (0.003) | 0.7408 (0.004) | **0.7418** (0.003) | 0.7395 (0.003) | 0.7394 (0.003) |
| DTLZ3 | **0.1680** (0.115) | 0.1680 (0.115) | 0.0000 (0.000) | 0.0018 (0.009) | 0.0069 (0.038) | 0.0167 (0.058) | 0.0063 (0.034) | 0.0000 (0.000) | 0.0204 (0.059) |
| DTLZ4 | 0.7051 (0.099) | 0.7390$^\dagger$ (0.028) | **0.7480**$^\dagger$ (0.002) | 0.7465$^\dagger$ (0.004) | 0.7466$^\dagger$ (0.003) | 0.7474$^\dagger$ (0.003) | 0.7463$^\dagger$ (0.002) | 0.7469$^\dagger$ (0.003) | 0.7471$^\dagger$ (0.003) |
| DTLZ5 | 0.2624 (0.001) | **0.2638**$^\dagger$ (0.001) | 0.2625 (0.001) | 0.2630 (0.001) | 0.2625 (0.001) | 0.2631$^\dagger$ (0.001) | 0.2629 (0.001) | 0.2630 (0.001) | 0.2630 (0.001) |
| DTLZ6 | 0.1106 (0.030) | **0.2604**$^\dagger$ (0.004) | 0.2143$^\dagger$ (0.016) | 0.2196$^\dagger$ (0.020) | 0.2275$^\dagger$ (0.014) | 0.2209$^\dagger$ (0.014) | 0.2202$^\dagger$ (0.023) | 0.2204$^\dagger$ (0.023) | 0.2135$^\dagger$ (0.030) |
| DTLZ7 | 0.5222 (0.028) | 0.5222 (0.028) | 0.5440$^\dagger$ (0.005) | 0.5454$^\dagger$ (0.004) | 0.5450$^\dagger$ (0.003) | 0.5451$^\dagger$ (0.004) | **0.5455**$^\dagger$ (0.004) | 0.5451$^\dagger$ (0.003) | 0.5453$^\dagger$ (0.004) |

Using a Wilcoxon rank sum test with 0.05 significance level, † indicates significantly better IGD than IBEA with its default operator (SBX). Boldface entries indicates best mean value.

# BIBLIOGRAPHY

[1] Planet Labs. https://www.planet.com/.

[2] The role of small satellites in NASA and NOAA Earth observation programs. Technical report, Space Studies Board National Research Council, 2000.

[3] Rakesh Agrawal, T Imielinski, and A Swami. Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, 22(May):207–216, 1993.

[4] Rakesh Agrawal and Ramakrishnan Srikant. Fast Algorithms for Mining Association Rules. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, 1994.

[5] Anne Auger, Johannes Bader, Dimo Brockhoff, and Eckart Zitzler. Theory of the hypervolume indicator: Optimal -Distributions and the Choice of the Reference Point. *Proceedings of the tenth ACM SIGEVO workshop on Foundations of genetic algorithms - FOGA '09*, page 87, 2009.

[6] Richard Balling. Design by shopping: A new paradigm? In *Proceedings of the Third World Congress of structural and multidisciplinary optimization (WCSMO-3)*, pages 295–297, 1999.

[7] Sunith Bandaru, Amos H. C. Ng, and Kalyanmoy Deb. Data Mining Methods for Knowledge Discovery in Multi-Objective Optimization: Part A - Survey. *Expert Systems with Applications*, 70:139–159, 2016.

[8] Sunith Bandaru, Amos H. C. Ng, and Kalyanmoy Deb. Data Mining Methods for Knowledge Discovery in Multi-Objective Optimization : Part B - New Developments and Applications. *Expert Systems With Applications*, 70:119–138, 2017.

[9] Sunith Bandaru, Amos H. C. Ng, and Kalyanmoy Deb. Data mining methods for knowledge discovery in multi-objective optimization: Part A - Survey. *Expert Systems with Applications*, 70:139–159, 2017.

[10] Jany Belluz, Saint Martin, Marco Gaudesi, Giovanni Squillero, Politecnico Torino, Corso Duca, Alberto Tonda, and Lucien Brétignères. Operator Selection using Improved Dynamic Multi-Armed Bandit Categories and Subject Descriptors. *Genetic and Evolutionary Computation Conference 2015*, pages 1311–1317, 2015.

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

[12] William J. Blackwell, D Burianek, K Clark, D Crompton, A Cunningham, L Fuhrman, P Hopman, and S Michael. The NASA TROPICS CubeSat Constellation Mission: Overview and Science Objectives. In *31st Annual AIAA/USU Conference on Small Satellites*, 2017.

[13] Piero P. Bonissone, Raj Subbu, Neil Eklund, and Thomas R. Kiehl. Evolutionary algorithms + domain knowledge = real-world evolutionary computation. *IEEE Transactions on Evolutionary Computation*, 10(3):256–280, 2006.

[14] Christopher R. Boshuizen, James Mason, Pete Klupar, and Shannon Spanhake. Results from the Planet Labs Flock Constellation. *28th Annual AIAA/USU Conference on Small Satellites*, pages SSC14–I–1, 2014.

[15] Peter A. N. Bosman and Dirk Thierens. The balance between proximity and diversity in multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 7(2):174–188, 2003.

[16] Dimo Brockhoff, Tobias Wagner, and Heike Trautmann. R2 Indicator-Based Multiobjective Search. *Evolutionary Computation*, 23(3):369–395, 2015.

[17] Edmund K. Burke, Michel Gendreau, Matthew R. Hyde, Graham Kendall, Gabriela Ochoa, Ender Özcan, and Rong Qu. Hyper-heuristics: a survey of the state of the art. *Journal of the Operational Research Society*, 64(12):1695–1724, jul 2013.

[18] J.C. Calvo, J. Ortega, and M. Anguita. PITAGORAS-PSP: Including domain knowledge in a multi-objective approach for protein structure prediction. *Neurocomputing*, 74(16):2675–2682, 2011.

[19] Caner Candan, Adrien Goeffon, Frédéric Lardeux, and Frédéric Saubion. A dynamic island model for adaptive operator selection. *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference - GECCO '12*, page 1253, 2012.

[20] Dirk G. Cattrysse and Luk N. Van Wassenhove. A survey of algorithms for the generalized assignment problem. *European Journal of Operational Research*, 60(3):260–272, 1992.

[21] Guido Cervone, Pasquale Franzese, and Allen P K Keesee. Algorithm quasi-optimal (AQ) learning. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(2):218–236, 2010.

[22] Timur Chabuk, James Reggia, Jason Lohn, and Derek Linden. Causally-guided evolutionary optimization and its application to antenna array design. *Integrated Computer-Aided Engineering*, 19(2):111–124, 2012.

[23] Chang Wook Ahn and R.S. Ramakrishna. On the Scalability of Real-Coded Bayesian Optimization Algorithm. *IEEE Transactions on Evolutionary Computation*, 12(3):307–322, jun 2008.

[24] Si Wei Cheng, Hui Zhang, Lin Cheng Shen, and Jing Chen. Optimization of regional coverage reconnaissance satellite constellation by improved NSGA-II algorithm. *Proceedings - International Conference on Intelligent Computation Technology and Automation, ICICTA 2008*, 1:660–664, 2008.

[25] Karim J. Chichakly and Margaret J. Eppstein. Discovering design principles from dominated solutions. *IEEE Access*, 1(iii):275–289, 2013.

[26] Po Wen Chiu and Christina Bloebaum. Hyper-Radial Visualization (HRV) for Decision-Making in Multi-Objective Optimization. In *46th AIAA Aerospace Sciences Meeting and Exhibit*, pages 1–16, 2008.

[27] Carlos A. Coello Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245–1287, 2002.

[28] Carlos A. Coello Coello, Gary B Lamont, and David A Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Genetic and Evolutionary Computation Series. Springer US, Boston, MA, 2nd edition, 2007.

[29] Benjamin A. Corbin. *The Value Proposition of Distributed Satellite Systems for Space Science Missions*. PhD thesis, Massachusetts Institute of Technology, 2015.

[30] Peter Cowling, Graham Kendall, and Eric Soubeiga. A hyperheuristic approach to scheduling a sales summit. *Practice and Theory of Automated Timetabling III*, pages 176–190, 2000.

[31] Edward F. Crawley, Bruce G. Cameron, and Daniel Selva. System Architecture: Strategy and Product Development for Complex Systems, 2016.

[32] Luis Da Costa, Álvaro Fialho, Marc Schoenauer, and Michèle Sebag. Adaptive operator selection with dynamic multi-armed bandits. *Genetic and Evolutionary Computation Conference*, 2008.

[33] Lawrence Davis. *Handbook of Genetic Algorithms.* Van Nostrand Reinhold, New York, New York, USA, 1 edition, 1991.

[34] Kalyanmoy Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineeringe*, 186:311–338, 2000.

[35] Kalyanmoy Deb and Ram Bhushan Agrawal. Simulated Binary Crossover for Continuous Search Space. Technical report, India Institute of Technology, Kanpur, UP, India, 1994.

[36] Kalyanmoy Deb, Ashish Anand, and Dhiraj Joshi. A computationally efficient evolutionary algorithm for real-parameter optimization. *Evolutionary computation*, 10(4):371–395, 2002.

[37] Kalyanmoy Deb, Sunith Bandaru, David Greiner, Antonio Gaspar-Cunha, and Cem Celal Tutum. An integrated approach to automated innovization for discovering useful design principles: Case studies from engineering. *Applied Soft Computing*, 15:42–56, 2014.

[38] Kalyanmoy Deb, Manikanth Mohan, and Shikhar Mishra. A fast multi-objective evolutionary algorithm for finding well-spread pareto-optimal solutions. Technical report, Indian Institute of Technology Kanpur, 2003.

[39] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[40] Kalyanmoy Deb and Aravind Srinivasan. Innovization: Innovating design principles through optimization. *GECCO '06 Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1629–1636, 2006.

[41] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable Test Problems for Evolutionary Multiobjective Optimization. In *Evo-*

*lutionary Multiobjective Optimization*, pages 105–145. Springer-Verlag, London, 2005.

[42] Marie-Jose Deutsch and Joy S. Nichols. Advanced approach to concept and design studies for space missions. *Astrophysics and Space Science*, 273:201–206, 2000.

[43] Alan Díaz-Manríquez, Gregorio Toscano-Pulido, Carlos A. Coello Coello, and Ricardo Landa-Becerra. A ranking method based on the R2 indicator for many-objective optimization. *2013 IEEE Congress on Evolutionary Computation, CEC 2013*, pages 1523–1530, 2013.

[44] John H Drake, Ender Özcan, and Edmund K Burke. A Case Study of Controlling Crossover in a Selection Hyper-heuristic Framework Using the Multidimensional Knapsack Problem. *Evolutionary Computation*, 24(1):113–141, mar 2016.

[45] Catarina Dudas, Amos H. C. Ng, and Henrik Boström. Post-analysis of multi-objective optimization solutions using decision trees. *Intelligent Data Analysis*, 19(2):259–278, 2015.

[46] Juan J. Durillo and Antonio J. Nebro. jMetal: A Java framework for multi-objective optimization. *Advances in Engineering Software*, 42(10):760–771, 2011.

[47] Morgan Dwyer, Daniel Selva, I.D. Portillo, M. Sanchez-Net, B. Cameron, Z. Szajnfarber, and Edward F. Crawley. Exploring the trade-offs of aggregated versus disaggregated architectures for environmental monitoring in low-earth orbit. *AIAA SPACE 2014 Conference and Exposition*, 2014.

[48] A E Eiben, Zbigniew Michalewicz, M Schoenauer, and J. E. Smith. Parameter Control in Evolutionary Algorithms. *Studies in Computational Intelligence*, 54:19–46, 2007.

[49] Emanuel Falkenauer. *Genetic Algorithms and Grouping Problems.* John Wiley & Sons Ltd., West Essex, England, 1 edition, 1998.

[50] Matthew P. Ferringer, Ronald S. Clifton, and Timothy G. Thompson. Constellation Design with Parallel Multi-Objective Evolutionary Computation. *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, pages 1–19, 2006.

[51] Matthew P. Ferringer, Ronald S. Clifton, and Timothy G. Thompson. Efficient and Accurate Evolutionary Multi-Objective Optimization Paradigms for Satellite Constellation Design. *Journal of Spacecraft and Rockets*, 44(3):682–691, 2007.

[52] Matthew P. Ferringer, Marc Diprinzio, Timothy Thompson, Kyle Hanifen, William R. Whittecar, and Patrick M. Reed. A Framework for the Discovery of Passive-Control, Minimum Energy Satellite Constellations. In *AIAA/AAS Astrodynamics Specialist Conference AIAA*, pages 1–24, 2014.

[53] Matthew P. Ferringer and David B Spencer. Satellite Constellation Design Tradeoffs Using Multiple-Objective Evolutionary Computation. *Journal of Spacecraft and Rockets*, 43(6), 2006.

[54] Matthew P. Ferringer, David B. Spencer, and Patrick M. Reed. Many-objective reconfiguration of operational satellite constellations with the large-cluster epsilon non-dominated sorting genetic algorithm-II. *2009 IEEE Congress on Evolutionary Computation, CEC 2009*, pages 340–349, 2009.

[55] Álvaro Fialho. *Adaptive Operator Selection for Optimization*. PhD thesis, Universite Paris Sud, 2010.

[56] Álvaro Fialho, Luis Da Costa, Marc Schoenauer, and Michèle Sebag. Extreme value based adaptive operator selection. In *Parallel Problem Solving From Nature*, pages 175–184, 2008.

[57] Álvaro Fialho, Luis Da Costa, Marc Schoenauer, and Michèle Sebag. Dynamic multi-armed bandits and extreme value-based rewards for adaptive operator selection. *Proc. LION-3. Springer-Verlag*, pages 176–190, 2009.

[58] Álvaro Fialho, Luis Da Costa, Marc Schoenauer, and Michèle Sebag. Analyzing bandit-based adaptive operator selection mechanisms. *Annals of Mathematics and Artificial Intelligence*, 60(1):25–64, 2010.

[59] Álvaro Fialho, Marc Schoenauer, and Michèle Sebag. Toward comparison-based adaptive operator selection. *Proceedings of the 12th annual conference on Genetic and evolutionary computation - GECCO '10*, page 767, 2010.

[60] M. L. Fisher, R. Jaikumar, and L. N. Van Wassenhove. A Multiplier Adjustment Method for the Generalized Assignment Problem. *Management Science*, 32(9):1095–1103, 1986.

[61] Robert S Fraser, Shana Mattoo, Eueng-nan Yeh, and C. R. McClain. Algorithm for atmospheric and glint corrections of satellite measurements of ocean pigment. *Journal of Geophysical Research: Atmospheres*, 102(D14):17107–17118, jul 1997.

[62] Peter I. Frazier and Jialei Wang. Bayesian Optimization for Materials Design. In *Springer Series in Materials Science*, volume 225, pages 45–75. Springer International Publishing Switzerland, 2016.

[63] Tobias Friedrich and Markus Wagner. Seeding the initial population of multi-objective evolutionary algorithms: A computational study. *Applied Soft Computing Journal*, 33:223–230, 2015.

[64] Salvador Garcia, Daniel Molina, Manuel Lozano, and Francisco Herrera. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 Special Session on Real Parameter Optimization. *Journal of Heuristics*, 15(6):617–644, 2009.

[65] Abhinav Gaur and Kalyanmoy Deb. Adaptive Use of Innovization Principles for a Faster Convergence of Evolutionary Multi-Objective Optimization Algorithms. Technical Report July, COIN Report Number 2016003, East Lansing, MI, 2016.

[66] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison Wesley, 1 edition, 1989.

[67] David E. Goldberg. Probability matching, the magnitude of reinforcement, and classifier system bidding. *Machine Learning*, 5(4):407–425, 1990.

[68] Richard A. Gonçalves, Carolina P Almeida, and Aurora Pozo. Upper Confidence Bound (UCB) Algorithms for Adaptive Operator Selection in MOEA/D. In *Evolutionary Multi-Criterion Optimization; 8th International Conference, EMO 2015*, volume 9018, pages 411–425, 2015.

[69] Richard A. Gonçalves, Josiel N. Kuk, Carolina P Almeida, and Sandra M Venske. MOEA/D-HH: A Hyper-Heuristic for Multi-objective Problems. In *Evolutionary Multi-Criterion Optimization; 8th International Conference, EMO 2015*, volume 9018, pages 94–108, 2015.

[70] Wenyin Gong, Álvaro Fialho, Zhihua Cai, and Hui Li. Adaptive strategy selection in differential evolution for numerical optimization: An empirical study. *Information Sciences*, 181(24):5364–5386, 2011.

[71] David M Hadka. MOEA Framework - A Free and Open Source Java Framework for Multiobjective Optimization. Version 2.12, 2017. http://www.moeaframework.org/

[72] David M Hadka and Patrick M. Reed. Diagnostic Assessment of Search Controls and Failure Modes in Many-Objective Evolutionary Optimization. *Evolutionary Computation*, 20(3):423–452, 2012.

[73] David M Hadka and Patrick M. Reed. Borg: an auto-adaptive many-objective evolutionary computing framework. *Evolutionary computation*, 21(2):231–59, jan 2013.

[74] William E. Hart, N Krasnogor, and J. E. Smith. *Recent Advances in Memetic Algorithms*, volume 166 of *Studies in Fuzziness and Soft Computing*. Springer-Verlag, Berlin/Heidelberg, 2005.

[75] Nozomi Hitomi, Hyunseung Bang, and Daniel Selva. Extracting and Applying Knowledge with Adaptive Knowledge-driven Optimization to Architect and Earth Observing Satellite System. *AIAA Information Systems at the AIAA SciTech Forum*, pages 1–21, 2017.

[76] Nozomi Hitomi and Daniel Selva. The effect of credit definition and aggregation strategies on multi-objective hyper-heuristics. In *ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2015.

[77] Nozomi Hitomi and Daniel Selva. A Hyperheuristic Approach To Leveraging Domain Knowledge In Multiobjective Evolutionary Algorithms. In *Proceedings of the ASME 2016 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2016*, 2016.

[78] Nozomi Hitomi and Daniel Selva. A Classification and Comparison of Credit Assignment Strategies in Multiobjective Adaptive Operator Selection. *IEEE Transactions on Evolutionary Computation*, 21(2):294–314, apr 2017.

[79] Nozomi Hitomi and Daniel Selva. Constellation Optimization Using an Evolutionary Algorithm with a Variable-length Chromosome. *IEEE Aerospace Conference*, 2018.

[80] Yu-Chi Ho and D.L. Pepyne. Simple Explanation of the No-Free-Lunch Theorem and Its Implications. *Journal of Optimization Theory and Applications*, 115(3):549–570, 2002.

[81] John H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, 1975.

[82] V. L. Huang, A. K. Qin, P. N. Suganthan, and M. F. Tasgetiren. Multi-objective optimization based on self-adaptive differential evolution algorithm. In *2007 IEEE Congress on Evolutionary Computation*, pages 3601–3608. IEEE, sep 2007.

[83] Simon Huband, Phil Hingston, Luigi Barone, and Lyndon While. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, 2006.

[84] A. Iorio and X. Li. Rotationally Invariant Crossover Operators in Evolutionary Multi-objective Optimization. *6th International Conference, SEAL 2006. Lecture Notes in Computer Science*, 4247:310–317, 2006.

[85] Hisao Ishibuchi, Naoya Akedo, and Yusuke Nojima. Relation between neighbourhood size and MOEA/D performance on many-objective problems. *Lecture Notes in Computer Science 7811: Evolutionary Multi-Criterion Optimization - EMO*, (March):459–474, 2013.

[86] Hisao Ishibuchi, Yasuhiro Hitotsuyanagi, Noritaka Tsukamoto, and Yusuke Nojima. Many-objective test problems to visually examine the behavior of multiobjective evolution in a decision space. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6239 LNCS(PART 2):91–100, 2010.

[87] Curtis Iwata, Samantha Infeld, Jennifer M. Bracken, Mellisa McGuire, Christina McQuirk, Aron Kisdi, Johnathan Murphy, Bjorn Cole, and Pezhman Zarifian. Model-Based Systems Engineering in Concurrent Engineering Centers. In *AIAA SPACE 2015 Conference and Exposition*, pages 1–13, 2015.

[88] Ralf Jahr, Horia Calborean, Lucian Vintan, and Theo Ungerer. Boosting design space explorations with existing or automatically learned knowledge. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7201:221–235, 2012.

[89] Sajjad M. Jasimuddin, Jonathan H. Klein, and Con Connell. The paradox of using tacit and explicit knowledge. *Management Decision*, 43(1):102–112, 2005.

[90] Cyrus D. Jilla and David W. Miller. Multi-Objective, Multidisciplinary Design Optimization Methodology for Distributed Satellite Systems. *Journal of Spacecraft and Rockets*, 41(1):39–50, 2004.

[91] Yaochu Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9(October 2003):3–12, 2005.

[92] Yaochu Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61–70, 2011.

[93] Mark D Johnston. An Evolution Algorithm Approach to Multi-objective Scheduling of Space Network Communications. *Intelligent Automation and Soft Computing*, 14(3):367–376, 2008.

[94] L. Jourdan, David W Corne, D. Savic, and G. Walters. Hybridising Rule Induction and Multi-Objective Evolutionary Search for Optimising Water Distribution Systems. In *Fourth International Conference on Hybrid Intelligent Systems (HIS'04)*, pages 434–439, 2004.

[95] Laetitia Jourdan, David W Corne, Dragan Savic, and Godfrey Walters. Preliminary Investigation of the Learnable Evolution Model' for Faster/Better Multiobjective Water Systems Design. In *Third International Conference, EMO 2005.*, pages 841–855, 2005.

[96] Laetitia Jourdan, Clarisse Dhaenens, and El-ghazali Talbi. Using Datamining Techniques to Help Metaheuristics: A Short Survey. In *Lecture Notes in Computer Science: Hybrid Metaheuristics*, pages 57–69. Springer-Verlag Berlin Heidelberg, 2006.

[97] Giorgos Karafotias, Mark Hoogendoorn, and A. E. Eiben. Parameter Control in Evolutionary Algorithms: Trends and Challenges. *IEEE Transactions on Evolutionary Computation*, 19(2):167–187, 2015.

[98] Borhan Kazimipour, Xiaodong Li, and A K Qin. A review of population initialization techniques for evolutionary algorithms. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 2585–2592. IEEE, jul 2014.

[99] H Kita, I Ono, and S Kobayashi. Multi-parental extension of the unimodal normal distribution crossover for real-coded genetic algorithms. *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99)*, pages 1581–1587, 1999.

[100] Nathan Knerr and Daniel Selva. Cityplot: Visualization of High-Dimensional Design Spaces With Multiple Criteria. *Journal of Mechanical Design*, 138(9):091403, jul 2016.

[101] Joshua D. Knowles. ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, 2006.

[102] Herbert J. Kramer. *Observation of the Earth and Its Environment*. Springer Berlin Heidelberg, Berlin, Heidelberg, 4th edition, 2002.

[103] Eduardo Krempser, Álvaro Fialho, and Helio J C Barbosa. Adaptive operator selection at the hyper-level. In *Parallel Problem Solving from Nature - PPSN 2012*, volume 7492, pages 378–387, 2012.

[104] Fumiya Kudo and Tomohiro Yoshikawa. Knowledge extraction in multi-objective optimization problem based on visualization of Pareto solutions. *2012 IEEE Congress on Evolutionary Computation*, pages 1–6, 2012.

[105] Marco Laumanns, Lothar Thiele, Kalyanmoy Deb, and Eckart Zitzler. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary computation*, 10(3):263–82, jan 2002.

[106] Armando M. Leite da Silva, Muriell R. Freire, and Leonardo M. Honório. Transmission expansion planning optimization by adaptive multi-operator evolutionary algorithms. *Electric Power Systems Research*, 133:173–181, 2016.

[107] Ke Li, Kalyanmoy Deb, Qingfu Zhang, and Sam Kwong. An Evolutionary Many-Objective Optimization Algorithm Based on Dominance and Decomposition. *IEEE Transactions on Evolutionary Computation*, 19(5):694–716, 2015.

[108] Ke Li, Álvaro Fialho, and Sam Kwong. Multi-Objective Differential Evolution with Adaptive Control of Parameters and Operators. *Lecture Notes in Computer Science 6683: Learning and Intelligent Optimization*, 6683:473–487, 2011.

[109] Ke Li, Álvaro Fialho, Sam Kwong, and Qingfu Zhang. Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 18(1):114–130, 2014.

[110] Miqing Li, Shengxiang Yang, and Xiaohui Liu. Shift-based density estimation for pareto-based algorithms in many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 18(3):348–365, 2014.

[111] Miqing Li, Shengxiang Yang, and Xiaohui Liu. Pareto or Non-Pareto: Bi-Criterion Evolution in Multi-Objective Optimization. *IEEE Transactions on Evolutionary Computation*, pages 1–21, 2015.

[112] Miqing Li, Shengxiang Yang, Xiaohui Liu, and Ruimin Shen. A comparative study on evolutionary algorithms for many-objective optimization. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7811 LNCS:261–275, 2013.

[113] Qiuzhen Lin, Zhiwang Liu, Qiao Yan, Zhihua Du, Carlos A. Coello Coello, Zhengping Liang, Wenjun Wang, and Jianyong Chen. Adaptive composite operator selection and parameter control for multiobjective evolutionary algorithm. *Information Sciences*, 339:332–352, 2016.

[114] Bing Liu, Wynne Hsu, Yiming Ma, and Blwhy Ma. Integrating Classification and Association Rule Mining. *Knowledge Discovery and Data Mining*, pages 80–86, 1998.

[115] Xiaolong (Luke) Zhang, Timothy W. Simpson, Mary Frecker, and George Lesieutre. Supporting knowledge exploration and discovery in multi-dimensional data with interactive multiscale visualisation. *Journal of Engineering Design*, 23(1):23–47, 2012.

[116] Khin Lwin, Rong Qu, and Graham Kendall. A learning-guided multi-objective evolutionary algorithm for constrained portfolio optimization. *Applied Soft Computing Journal*, 24:757–772, 2014.

[117] Mashael Maashi, Graham Kendall, and Ender Özcan. Choice function based hyper-heuristics for multi-objective optimization. *Applied Soft Computing Journal*, 28:312–326, 2015.

[118] Md Shahriar Mahbub, Markus Wagner, Luigi Crema, and Fondazione Bruno Kessler. Incorporating domain knowledge into the optimization of energy systems. *Applied Soft Computing*, 47:483–493, 2016.

[119] Mark W. Maier and Eberhardt Rechtin. *The Art of Systems Architecting*. CRC press, Boca Raton, FL, 3 edition, 2009.

[120] Rammohan Mallipeddi and Ponnuthurai Nagaratnam Suganthan. Differential evolution algorithm with ensemble of parameters and mutation and crossover strategies. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6466 LNCS:71–78, 2010.

[121] Jorge Maturana, Álvaro Fialho, Frédéric Saubion, Marc Schoenauer, and Michèle Sebag. Extreme compass and Dynamic Multi-Armed Bandits for Adaptive Operator Selection. *2009 IEEE Congress on Evolutionary Computation*, pages 365–372, 2009.

[122] Jorge Maturana, Frédéric Lardeux, and Frédéric Saubion. Autonomous operator management for evolutionary algorithms. *Journal of Heuristics*, 16(6):881–909, mar 2010.

[123] Jorge Maturana and Frédéric Saubion. A compass to guide genetic algorithms. *Parallel Problem Solving from NaturePPSN X*, pages 256–265, 2008.

[124] Helmut Mayer. Air pollution in cities. *Atmospheric Environment*, 33(24-25):4029–4037, 1999.

[125] K McClymont and Ed.C. Keedwell. Markov chain hyper-heuristic (MCHH): An online selective hyper-heuristic for multi-objective continuous problems. *Genetic and Evolutionary Computation Conference, GECCO'11*, pages 2003–2010, 2011.

[126] Bertolt Meyer and Kozo Sugiyama. The concept of knowledge in KM: a dimensional model. *Journal of Knowledge Management*, 11(1):17–35, feb 2007.

[127] I. Meziane-Tani, G. Métris, G. Lion, A. Deschamps, F. T. Bendimerad, and M. Bekhti. Optimization of small satellite constellation design for continuous mutual regional coverage with multi-objective genetic algorithm. *International Journal of Computational Intelligence Systems*, 9(4):627–637, 2016.

[128] Efrén Mezura-Montes and Carlos A. Coello Coello. Constraint-handling in nature-inspired numerical optimization: Past, present and future. *Swarm and Evolutionary Computation*, 1(4):173–194, 2011.

[129] Zbigniew Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag Berlin Heidelberg, 3rd edition, 1996.

[130] Zbigniew Michalewicz. Repair algorithms. In David B . Fogel, Thomas Back, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, chapter C5.4, pages 2–6. IOP Publishing Ltd, 1997.

[131] Ryszard S. Michalski. On the Quasi-Minimal Solution of the General Covering Problem, 1969.

[132] Ryszard S. Michalski. LEARNABLE EVOLUTION MODEL: Evolutionary Processes Guided by Machine Learning. *Machine Learning*, 38(1):9–40, 2000.

[133] Ryszard S. Michalski, Guido Cervone, and Kenneth Kaufman. Speeding up evolution through learning: LEM. *Advances in Soft Computing*, 4:243–256, 2000.

[134] M. Misir, K. Verbeeck, P. De Causmaecker, and G. Vanden Berghe. The effect of the set of low-level heuristics on the performance of selection hyper-heuristics. In *Parallel Problem Solving from Nature - PPSN XII*, volume 7492 LNCS, pages 408–417, 2012.

[135] Sreeja Nag, Charles K. Gatebe, and Thomas Hilker. Simulation of Multi-angular Remote Sensing Products Using Small Satellite Formations. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(2):638–653, 2016.

[136] Sreeja Nag, Charles K. Gatebe, David W. Miller, and Olivier L. De Weck. Effect of satellite formations and imaging modes on global albedo estimation. *Acta Astronautica*, 126:77–97, 2016.

[137] Sreeja Nag, Steven P Hughes, and Jacqueline Le Moigne. Streamlining the Design Tradespace for Earth Imaging Constellations. In *AIAA SPACE 2016*, pages 1–17, Reston, Virginia, sep 2016. American Institute of Aeronautics and Astronautics.

[138] Sreeja Nag, Jacqueline LeMoigne, David W. Miller, and Olivier L. de Weck. A framework for orbital performance evaluation in Distributed Space Missions for earth observation. In *2015 IEEE Aerospace Conference*, pages 1–20. IEEE, mar 2015.

[139] National Research Council. Earth Science and Applications from Space. Technical report, National Research Council, Washington, D.C., oct 2007.

[140] Amos H. C. Ng, Catarina Dudas, Henrik Boström, and Kalyanmoy Deb.

Interleaving Innovization with Evolutionary Multi-Objective Optimization in Production System Simulation for Faster Convergence. In G. Nicosia and P. Pardalos, editors, *7th International Conference, LION 7, LNCS*, volume 7997, pages 1–18. Springer-Verlag Berlin Heidelberg, 2013.

[141] Shigeru Obayashi and Daisuke Sasaki. Visualization and Data Mining of Pareto Solutions Using Self-Organizing Map. *Evolutionary Multi-Criterion Optimization 2003 LNCS*, 2632:796–809, 2003.

[142] Robert E Oberto, Erik Nilsen, Ron Cohen, Rebecca Wheeler, Paul DeFlorio, and Chester Borden. The NASA Exploration Design Team: Blueprint for a New Design Paradigm. *IEEE Aerospace Conference*, page 8, 2005.

[143] M. Gregory O'Neill and Annalisa L. Weigel. Assessing Fractionated Spacecraft Value Propositions for Earth Imaging Space Missions. *Journal of Spacecraft and Rockets*, 48(6):974–986, 2011.

[144] I Ono, H Kita, and S Kobayashi. A real-coded genetic algorithm using the unimodal normal distribution crossover. *Advances in Evolutionary Computing*, pages 213–237, 2003.

[145] Ender Özcan, Yuri Bykov, Murat Birben, and Edmund K. Burke. Examination timetabling using late acceptance hyper-heuristics. *2009 IEEE Congress on Evolutionary Computation, CEC 2009*, pages 997–1004, 2009.

[146] Martin Pelikan, David E. Goldberg, and Erick Cantú-Paz. BOA: The Bayesian Optimization Algorithm. *Genetic and Evolutionary Computation*, 1:525–532, 1999.

[147] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

[148] Dung H. Phan and Junichi Suzuki. R2-IBEA: R2 indicator based evolutionary algorithm for multiobjective optimization. *IEEE Congress on Evolutionary Computation*, pages 1836–1845, 2013.

[149] J. Puig-Suari, C. Turner, and W. Ahlgren. Development of the standard CubeSat deployer and a CubeSat class\nPicoSatellite. *2001 IEEE Aerospace Conference Proceedings (Cat. No.01TH8542)*, 1:347–353, 2001.

[150] R.C. Purshouse and P.J. Fleming. On the evolutionary optimisation of many conflicting objectives. *IEEE Transactions on Evolutionary Computation*, 11(6):770–784, 2007.

[151] A K Qin and P N Suganthan. Self-adaptive differential evolution algorithm for numerical optimization. *IEEE Congress on Evolutionary Computation*, pages 1785–1791, 2005.

[152] Nicholas J. Radcliffe. Non-Linear Genetic Representations. *Parallel Problem Solving from Nature 2"*,, 2:259–268, 1992.

[153] Patrick M. Reed, Nathaniel W. Chaney, Jonathan D. Herman, Matthew P. Ferringer, and Eric F. Wood. Internationally coordinated multi-mission planning is now critical to sustain the space-based rainfall observations needed for managing floods globally. *Environmental Research Letters*, 10(2):024010, 2015.

[154] Randy Rose, Scott Gleason, and Christopher Ruf. NASA CYGNSS Mission Update; A Pathfinder for Operational GNSS Scatterometry Remote Sensing Applications. In *31st Annual AIAA/USU Conference on Small Satellites*, pages 1–10, 2017.

[155] Adam M. Ross and Daniel E. Hastings. The Tradespace Exploration Paradigm. *INCOSE International Symposium*, 15(1):1706–1718, 2005.

[156] C. Ruf, M. Unwin, J. Dickinson, R. Rose, D. Rose, M. Vincent, and A. Lyons. CYGNSS: Enabling the Future of Hurricane Prediction. *IEEE Geoscience and Remote Sensing Magazine*, 1(2):52–67, 2013.

[157] Sancho Salcedo-Sanz. A survey of repair methods used as constraint handling techniques in evolutionary algorithms. *Computer Science Review*, 3(3):175–192, 2009.

[158] Marc Sanchez, Daniel Selva, Bruce Cameron, Edward Crawley, Antonios Seas, and Bernie Seery. Results of the MIT Space Communication and Navigation architecture study. *2014 IEEE Aerospace Conference*, pages 1–14, mar 2014.

[159] Rainer Sandau. Status and trends of small satellite missions for Earth observation. *Acta Astronautica*, 66(1-2):1–12, 2010.

[160] M. G. Schlax, D. B. Chelton, and M. H. Freilich. Sampling errors in wind

fields constructed from single and tandem scatterometer datasets. *Journal of Atmospheric and Oceanic Technology*, 18(6):1014–1036, 2001.

[161] Daniel Selva. *Rule-based system architecting of Earth observation satellite systems*. PhD thesis, MIT, 2012.

[162] Daniel Selva. Experiments in Knowledge-intensive System Architecting : Interactive Architecture Optimization. *IEEE Aerospace Conference*, 2014.

[163] Daniel Selva. Knowledge-Intensive Global Optimization of Earth Observing System Architectures. In *SPIE Remote Sensing Conference*, 2014.

[164] Daniel Selva, Clara Abello, and Nozomi Hitomi. Preliminary experiments with learning agents in an interactive multi-agent systems architecture tradespace exploration tool. In *2015 Annual IEEE Systems Conference (SysCon) Proceedings*, pages 445–452. IEEE, apr 2015.

[165] Daniel Selva, Bruce G. Cameron, and Edward F. Crawley. Rule-Based System Architecting of Earth Observing Systems: Earth Science Decadal Survey. *Journal of Spacecraft and Rockets*, 51(5):1505–1521, sep 2014.

[166] Daniel Selva and David Krejci. A Survey and Assessment of the Capabilities of Cubesats for Earth Observation. *Acta Astronautica*, 74:50–68, 2012.

[167] Daniel Selva and David Krejci. Preliminary Assessment of Performance and Cost of a Cubesat Component of the Earth Science Decadal Survey. In *27th Annual AIAA/USU Conference on Small Satellites*. Conference on Small Satellites, 2013.

[168] Yann Semet and Marc Schoenauer. On the Benefits of Inoculation, an Example in Train Scheduling. *Proceedings of the 8th annual conference on Genetic and evolutionary computation GECCO 06*, page 1761, 2006.

[169] Alice E. Smith, David W. Coit, and Zbigniew Michalewicz. Penalty Functions. In Thomas Baeck, David Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, chapter 5, pages 1–6. IOP Publishing Ltd, 1997.

[170] Jorge A. Soria-Alcaraz, Gabriela Ochoa, Adrien Göeffon, Frédéric Lardeux, and Frédéric Saubion. Combining Mutation and Recombination to Improve a Distributed Model of Adaptive Operator Selection. In *International Conference on Artificial Evolution*, pages 97–108, 2016.

[171] Rainer Storn and K Price. Differential evolutiona simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, pages 341–359, 1997.

[172] Patrick D Surry and Nicholas J. Radcliffe. Inoculation to initialize evolutionary search. *Evolutionary Computing: AISB workshop*, 1143:269–285, 1996.

[173] Dirk Thierens. An adaptive pursuit strategy for allocating operator probabilities. *Belgian/Netherlands Artificial Intelligence Conference*, pages 385–386, 2005.

[174] Dirk Thierens. Adaptive Strategies for Operator Allocation. In Fernando G. Lobo, Cláudio F. Lima, and Zbigniew Michalewicz, editors, *Parameter Setting in Evolutionary Algorithms, Studies in Computational Intelligence*, volume 54 of *Studies in Computational Intelligence*, pages 77–90. Springer Berlin Heidelberg, Berlin, Heidelberg, vol. 54 edition, 2007.

[175] Dirk Thierens. Adaptive Operator Selection for Iterated Local Search. In *Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics: Second International Workshop, SLS 2009, Brussels.*, pages 140–144, 2009.

[176] S Tsutsui, M Yamamura, and T Higuchi. Multi-parent recombination with simplex crossover in real coded genetic algorithms. *Proceedings of the 1999 Genetic and Evolutionary Computation Conference (GECCO'99)*, pages 657–664, 1999.

[177] Dawei Van, Cong Liu, and Peng You. Multi-objective Optimization Design of Extended Walker Constellation for Global Coverage Services. In *IEEE International Conference on Computer and Communications*, pages 1309–1313, 2016.

[178] Jasper A. Vrugt and Bruce A. Robinson. Improved evolutionary optimization from genetically adaptive multimethod search. *Proceedings of the National Academy of Sciences of the United States of America*, 104(3):708–711, 2007.

[179] Tobias Wagner, Nicola Beume, and Boris Naujoks. Pareto-, aggregation-, and indicator-based methods in many-objective optimization. *Evolutionary MultiCriterion Optimization*, 4403:742–756, 2007.

[180] Shinya Watanabe, Yuta Chiba, and Masahiro Kanazaki. A proposal on analysis support system based on association rule analysis for non-dominated

solutions. *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*, pages 880–887, 2014.

[181] James R. Wertz, David F. Everett, and Jeffery J. Puschell, editors. *Space MIssion Engineering: The New SMAD*. The Microcosom Press, Hawthorne, CA, 2015.

[182] Lyndon While, Lucas Bradstreet, and Luigi Barone. A fast way of calculating exact hypervolumes. *IEEE Transactions on Evolutionary Computation*, 16(1):86–95, 2012.

[183] William R. Whittecar and Matthew P. Ferringer. Global Coverage Constellation Design Exploration Using Evolutionary Algorithms. In *AIAA/AAS Astrodynamics Specialist Conference*, pages 1–20, 2014.

[184] Edwin A. Williams, William A. Crossley, and Thomas J. Lang. Average and maximum revisit time trade studies for satellite constellations using a multiobjective genetic algorithm. In *Journal of the Astronautical Sciences*, volume 49.3, pages 385–400, 2001.

[185] Y G Woldesenbet, G G Yen, and B G Tessema. Constraint Handling in Multiobjective Evolutionary Optimization. *Evolutionary Computation, IEEE Transactions on*, 13(3):514–525, 2009.

[186] David H. Wolpert. No free lunch theorems for search. Technical report, Santa Fe Institute, Santa Fe, NM, 1995.

[187] David H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, apr 1997.

[188] Matthew J. Woodruff. *MANY OBJECTIVE VISUAL ANALYTICS : DECISION AIDING TOOLS FOR CONCEPTUAL DESIGN*. PhD thesis, Pennsylvania State University, 2016.

[189] Matthew J. Woodruff, Patrick M. Reed, and Timothy W. Simpson. Many objective visual analytics: rethinking the design of complex engineered systems. *Structural and Multidisciplinary Optimization*, 48(1):201–219, feb 2013.

[190] Matthew J. Woodruff and Timothy W Simpson. Problem Exploration With Many-Objective Visual Analytics. In *Proceedings of the ASME 2016 In-*

*ternational Design Engineering Technical Conferences and Computers and Information in Engineering Conference 2016*, pages 1–11, 2016.

[191] Matthew J. Woodruff, Timothy W Simpson, and Patrick M. Reed. Multi-objective evolutionary algorithms' performance in a support role. *Proceedings of the ASME 2015 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2015*, pages 1–12, 2015.

[192] T Yoshida and T Yoshikawa. A study on Non-Correspondence in Spread between objective space and design variable space for trajectory designing optimization problem. *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pages 2444–2450, 2014.

[193] Qingfu Zhang and Hui Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[194] Qingfu Zhang, Wudong Liu, and Hui Li. The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances. *2009 IEEE Congress on Evolutionary Computation, CEC 2009*, (1):203–208, 2009.

[195] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Wudong Liu. Multiobjective optimization Test Instances for the CEC 2009 Special Session and Competition. *Technical Report CES-487, The Shool of Computer Science and Electronic Engineering, University of Essex, Tech. Rep.*, pages 1–30, 2008.

[196] Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Qingfu Zhang. Decomposition-Based Multiobjective Evolutionary Algorithm with an Ensemble of Neighborhood Sizes. *IEEE Transactions on Evolutionary Computation*, 16(3):442–446, 2012.

[197] Aimin Zhou, Qingfu Zhang, and Yaochu Jin. Approximating the set of pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm. *IEEE Transactions on Evolutionary Computation*, 13(5):1167–1189, 2009.

[198] John Ziemer, Joan Ervin, and Jared Lang. Exploring Mission Concepts with the JPL Innovation Foundry A-Team. *Proceedings of the AIAA SPACE Conference*, pages 1–13, 2013.

[199] Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization : Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology Zurich, 1999.

[200] Eckart Zitzler, Joshua D. Knowles, and Lothar Thiele. Quality assessment of pareto set approximations. *Multiobjective Optimization: Interactive and Evolutionary Approaches*, 5252 LNCS:373–404, 2008.

[201] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, pages 95–100, 2001.

[202] Eckart Zitzler and K. Simon. Indicator-Based Selection in Multiobjective Search. *8th International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, pages 832–842, 2004.