Multi-Objective Factored Evolutionary Optimization and the Multi-Objective Knapsack Problem

Amy Peerlinck and John Sheppard
Gianforte School of Computing
Montana State University
Bozeman, MT, USA
amy.peerlinck@student.montana.edu, john.sheppard@montana.edu

***Abstract*—We propose a factored evolutionary framework for multi-objective optimization that can incorporate any multi-objective population based algorithm. Our framework, which is based on Factored Evolutionary Algorithms, uses overlapping subpopulations to increase exploration of the objective space; however, it also allows for the creation of distinct subpopulations as in co-operative co-evolutionary algorithms (CCEA). We apply the framework with the Non-Dominated Sorting Genetic Algorithm-II (NSGA-II), resulting in Factored NSGA-II. We compare NSGA-II, CC-NSGA-II, and F-NSGA-II on two different versions of the multi-objective knapsack problem. The first is the classic binary multi-knapsack implementation introduced by Zitzler and Thiele, where the number of objectives equals the number of knapsacks. The second uses a single knapsack where, aside from maximizing profit and minimizing weight, an additional objective tries to minimize the difference in weight of the items in the knapsack, creating a balanced knapsack. We further extend this version to minimize volume and balance the volume. The proposed 3-to-5 objective balanced single knapsack problem poses a difficult problem for multi-objective algorithms. Our results indicate that the non-dominated solutions found by F-NSGA-II tend to cover more of the Pareto front and have a larger hypervolume.**

***Index Terms*—multi-objective combinatorial optimization, co-operative coevolution, non-dominated sorting genetic algorithm, multi-objective knapsack**

I. Introduction

As multi-objective optimization (MOO) is becoming more prevalent in real world applications, the need for an adaptable approach to solve such problems becomes more apparent. Numerous approaches to solve MOO problems have been proposed, ranging from combinatorial optimization to engineering specific approaches [1], [2], [3]. However, as with many optimization problems, there is no one "best" solution to approach MOO [4]. But there is a general tendency in MOO research to use population-based approaches, generally termed multi-objective evolutionary algorithms (MOEA's). Surveys and comparative studies indicate that different MOEA's provide different benefits, but they also find that there is increasing difficulty in exploring the entire objective space as the number of objectives increases [5]. Reference-based approaches address this problem by guiding the algorithm in the search space using either automatically generated and evenly spread reference points/vectors, or based on user preference or prior knowledge of the problem [5].

We propose an approach to increase exploration of the objective space that can be used with any MOEA: the Multi-Objective Factored Evolutionary Algorithm (MOFEA). MOFEA is a framework that divides the population into subpopulations with overlapping variables; it uses the chosen MOEA to optimize the subpopulations and combines the resulting sets of non-dominated solutions through competition and sharing. MOFEA is an extension of the FEA framework introduced by Strasser *et al.* [6]. Furthermore, the use of subpopulations lends itself to parallelization and can thus be used to reduce computation time, which is desirable when dealing with large-scale MOO problems.

A popular benchmark problem that relates to many real world applications is the Multi-Objective 0-1 Knapsack (MO-KS) problem [7]. This problem was proposed as a benchmark for testing multi-objective combinatorial optimization (MOCO) algorithms. It makes for a good benchmark since it is easily altered in terms of number of objectives, constraints, and variables. However, increasing the number of knapsacks does not necessarily relate to real-world application [8]. To this end, a different multi-objective knapsack problem exists that uses a single knapsack but minimizes the difference in resources (e.g. weight) in the knapsack, creating a balanced knapsack [9].

In this paper, we utilize the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [10] as the underlying optimizer in the proposed MOFEA framework. NSGA-II is a popular method that has been shown to work well on problems with up to 3 objectives but starts declining in performance when the objectives increase further. The same authors then proposed NSGA-III, which uses the aforementioned reference directions to increase performance [11]. Carvalho and Britto found that the chosen reference points can positively or negatively impact the results found by NSGA-III, indicating that an automatically initialized set of reference directions may not be a desirable approach [12]. Furthermore, since we are initializing our problem instances randomly, we have no knowledge on where to place reference points on the hyperplane. Because of this, and in order to more clearly show the benefit of the MOFEA framework, we decided to use NSGA-II instead of NSGA-III in our experiments. Another popular technique that can be applied using many different population-based algorithms is the class of co-operative co-evolutionary algorithms (CCEA) [13].

CCEA creates disjoint subpopulations instead of overlapping ones and is the inspiration for the original FEA framework. Since CCEA's are also popular for solving MOO problems [13], we include a version of CCEA using NSGA-II in our experiments as well.

In the next section we give background information on multi-objective optimization, NSGA-II, and CCEA. Then, we cover relevant related work before providing details on the proposed MOFEA framework. In our experimental approach we include hypotheses, implementation details, chosen hyper-parameter settings, and chosen evaluation metrics. Section VI presents and discusses the results of the experiments. Finally, we present our conclusions and directions for future work.

II. Multi-Objective Optimization

Multi-Objective Optimization (MOO) is the process of optimizing more than one objective simultaneously [14]. Formally, without loss of generality, assume we wish to minimize k objectives. Then MOO consists of solving

$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathcal{X}}{\text{argmin}}\ \mathbf{f}(\mathbf{x})$$

where

$$\mathbf{f}(\mathbf{x}) = \{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})\}$$

with $k \geq 2$ conflicting objective functions $f_i : \mathbb{R}^n \to \mathbb{R}$, and $f_i \in F^k$ where F^k represents the objective space. Furthermore, $\mathbf{x} = [x_1, x_2, \ldots, x_n]^\top$ denotes an instantiation of the decision variables, where $\mathcal{X} \in \mathbb{R}^n$ is the solution space, and $\mathbf{x} \in \mathcal{X}$.

One common approach to solving MOO problems is to transform the k objectives into a single objective, enabling the application of single-objective methods. For example, if one of the objectives is known to be more important, an algorithm can simply optimize said objective. However, there are often competing objectives that hold similar importance. This leads to the following transformative approaches: aggregating the objectives using a weighted sum, or converting all objectives except one into constraints, where each transformed objective f_i is constrained by the worst value ϵ_i it is allowed to take. The latter is known as the ϵ-constraint method.

A lot of work in MOO focuses on continuous optimization; however, in this paper, we focus on a combinatorial optimization problem. This specific type of MOO is known as Multi-Objective Combinatorial Optimization (MOCO), which is interesting given many single-objective combinatorial optimization problems have been proven to be NP-hard [15]. Many different approaches have been proposed to address MOCO; some of which propose exact methods to solve two objective problems. However, such exact methods cannot be generalized to solve different problems or to handle more than two objectives. In order to solve problems in a more general manner, meta-heuristic algorithms have become a widespread approach with promising results [1]. There are two general classes of meta-heuristic approaches applied to MOCO: local search in the objective space and population-based search [16]. There are also many approaches that have been adapted to work for specific combinatorial problems but are not applicable to MOCO in general [17], [18].

A. Pareto Optimality

Pareto-based approaches have become the norm in multi-objective optimization where a set of optimal solutions is returned rather than attempting to find a single solution [19], [20]. Such approaches try to find Pareto non-dominated solutions, where Pareto dominance is defined as follows:

Definition 1: A point $\mathbf{x}^* \in \mathcal{X}$ is Pareto-optimal (or Pareto-dominant) if $\forall f_i \in \mathbf{f}, \forall \mathbf{x} \in \mathcal{X}, \mathbf{x} \neq \mathbf{x}^*, f_i(\mathbf{x}^*) \leq f_i(\mathbf{x})$, and $\exists f_j \in \mathbf{f}, f_j(\mathbf{x}^*) < f_j(\mathbf{x})$.

The implication of this definition is that, when improving an objective component of an "optimal" solution, at least one other component will be degraded.

This leads to the creation of a Pareto optimal set and Pareto optimal front:

Definition 2: The Pareto optimal set ($\mathbf{PS}^*$) is the set of non-dominated solutions with respect to the solution space $\mathcal{X}$.

Definition 3: The Pareto optimal front ($\mathbf{PF}^*$) is the set of points mapped from the Pareto optimal set onto the objective space F^k to form the boundary of the set of non-dominated solutions.

MOO-focused algorithms that use Pareto techniques try to find the Pareto optimal front, where the resulting Pareto front is called the approximate Pareto front.

B. Non-Dominated Sorting Genetic Algorithm-II

NSGA-II is one of the most popular Pareto-based algorithms in MOO and has been applied successfully to many common MOCO problems, e.g., travelling salesperson, knapsack, vehicle routing, and job shop scheduling [20]. NSGA was introduced by Srinivas *et al.* in 1994 [21], and improved in 2002 by Deb *et al.* to create NSGA-II [10]. NSGA-II is an elitist GA that finds Pareto non-dominated solutions and uses a crowding distance measure to maintain diversity in the next generation. Figure 1 shows how a population is sorted to perform selection for the next generation. The parent population P_t and the offspring population Q_t are combined into one population $R_t = P_t \cup Q_t$. R_t is sorted based on the non-domination principle, and individuals are assigned to different non-domination sets based on how good the solution is. If an entire set of non-dominated solutions is larger than the remaining slots for the next population, a second elimination is performed for that set based on crowding distance. The crowding distance measure is based on the cardinality of the solution set and its distance to the solution boundary.

C. Co-operative Co-Evolutionary MOO Algorithms

Cooperative co-evolutionary algorithms divide the population into subpopulations; these subpopulations represent a part of the solution, and are optimized separately from one another before being joined together again to represent the full solution [22]. However, subpopulations can also be defined to represent a single objective, thus optimizing each subpopulation along the separate objectives. The combination of CCEA's and

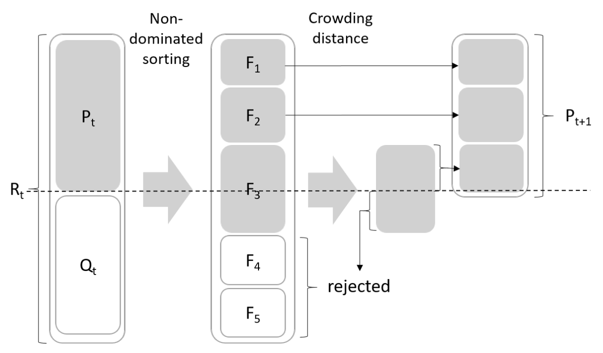


Fig. 1: NSGA-II selection procedure [10].

MOEA's is called co-operative co-evolutionary multi-objective optimisation algorithms (CCMOOA) [23]. The first combination of the Multi-Objective Genetic Algorithm (MOGA) [24] and CCEA [22] was presented by Keerativuttitumrong *et al.* [25]. They found that a co-operative approach can have beneficial results for finding a well spread out Pareto front when compared to the single population MOGA. This finding was confirmed in a follow-up study, where CCEA was applied to four different base algorithms (MOGA, NSGA-II, a controlled elitist NSGA, and a niched Pareto genetic algorithm), and compared to their single population alternatives [23].

III. Related Work

Several papers have been published that describe novel meta-heuristic approaches that aim to solve the Multi-Objective Knapsack problem. We focus our discussion on papers published in recent years, paying special attention to papers using a co-evolutionary approach. Zouache *et al.* propose a novel "cooperative" swarm intelligence algorithm for MOO; however, the term "cooperative" here does not refer to CCEA, but to the combination of the firefly algorithm with particle swarm optimization (MOFPA) [26]. They apply their approach to a knapsack with 250, 500, and 750 items, optimizing 2,3, and 4 objectives, resulting in $3\times3 = 9$ different knapsack problems. Their results indicate that their proposed hybrid algorithm performs better in terms of coverage on all instances of the knapsack problem studied when compared to NSGA-II, MOEA/D, and SPEA-II. However, the inverse generational distance metric, which compares the found Pareto front to the known optimal Pareto front, is not significantly different across any of the algorithms.

Mansour *et al.* split the population into subpopulations that are optimized in parallel using different configurations of their cooperative multi-objective local search algorithm based on weighted epsilon (Wϵ-CMOLS) [27]. They apply Wϵ-CMOLS to the 250 and 500 item knapsack with 2,3 and 4 objectives and compare it to NSGA-II, SPEA-II, Indicator Based Local Search Algorithm, Indicator Based Evolutionary Algorithm, and Pareto Local Search-Ant Colony Optimization. They find that their parallellization technique increases computational efficiency without strongly affecting the hypervolume results.

In an adjustment of the NSGA-III approach, Sahinkoc and Bilke use a fixed hyperplane and use subpopulations to evolve along the different objectives [28]. The fixed hyperplane is introduced to address the problem of the evenly spread reference points guiding the solutions in the wrong direction. To accomplish this, an optimal solution for each single objective is calculated and used as the fixed edge points of the hyperplane. They evaluate their method on the many-objective knapsack, solving a 500 item knapsack problem with 6, 8, 10, 15, 20, and 30 objectives. They find that including a fixed hyperplane significantly improves results for all NSGA-III implementations. The co-operative approach evolving along the different objectives improves the results further, and their proposed algorithm has the best performance on instances with a large number of objectives.

So far, the discussed literature all uses the Knapsack implementation as proposed by Zitzler and Thiele [7]. To the authors' knowledge, only one paper has explored the load balancing knapsack implementation. Specifically, Luo *et al.* propose a Pareto evolutionary algorithm based on Incremental Learning (PEAIL) and use the load balancing knapsack problem for their test case [8]. Incremental learning is an online learning technique that extracts historical information on the search behavior and feeds this information back to the evolutionary framework. Their algorithm also includes a competitive aspect between two of the three populations that are generated at different stages of the algorithm to improve convergence. The authors compare their algorithm to NSGA-II and find that PEAIL outperforms NSGA-II in terms of diversity, convergence, and dominance.

IV. Multi-Objective Factored Evolutionary Algorithms

Classic CCEA only creates subpopulations that have disjoint variables, i.e., there is no variable overlap between subpopulations. Strasser *et al.* proposed including overlap in subpopulations to create the Factored Evolutionary Algorithm (FEA), which has been shown to perform well on single-objective combinatorial optimization problems, such as NK-landscapes [6] and Bayesian network abductive inference [29]. Due to the overlap, FEA actually combines principles from both cooperative and competitive co-evolution by having subpopulations compete for representation of a decision variable and sharing discovered global solutions, where the term global solution refers to the full solution representing all decision variables.

We apply the concept of FEA to MOO by changing the compete and share steps of FEA to allow for tracking of a Pareto non-dominated population. We believe that using overlapping subpopulations will improve exploration by maintaining diversity: if a single decision variable is represented by two subpopulations, each population is able to explore a different part of the space that the decision variable is a part of, where the direction of search is only influenced by the other decision variables in the subpopulation. Furthermore, just as the original FEA allows using any population-based method, our adjustment of FEA allows the use of any MOEA; we denote this generalized algorithm MOFEA and show the pseudocode in Algorithm 1.

Since we are trying to build an approximate Pareto front, MOFEA keeps a set of global solutions as well as an archive of all non-dominated solutions. Initially, each subpopulation is assigned the same global solution to evaluate the total fitness of its individuals; however, as the algorithm proceeds, a random global solution out of the solution set is chosen for each subpopulation. Each subpopulation is optimized using an MOEA of choice, which returns a set of non-dominated solutions $\mathcal{N}''$. During the "Compete" step (lines 5–17), overlapping subpopulations use the non-dominated solution in $\mathcal{N}''$ with the best solution according to the criteria of the chosen MOEA, in the case of NSGA-II this is the non-dominated sorting principle, to represent the current decision variable (called "select_best"). Each potential solution for every decision variable is then saved in a temporary solution set $\mathcal{N}'$. The original chosen solution is also added to the temporary solution set. Once each variable and its representative populations have been examined, $\mathcal{N}'$ is evaluated to identify non-dominated solutions, which form the new set of global solutions and are added to the non-dominated archive. The new global solution set is then used to assign a global solution X randomly for each subpopulation, where the worst solution, according to the selection procedure of the implemented MOEA, in the subpopulation is replaced by X, completing the "Share" step (lines 19–22). This is one iteration of MOFEA. The algorithm repeats until a stopping criterion is met, for example a set number of iterations or the lack of change in the non-dominated solution set size or any other indicator.

Algorithm 1: Multi-Objective Factored Evolutionary Algorithm

```
input      : number of variables $m$, population size $n$,
             MOEA iterations $it$, number of objectives $k$,
             objective functions $F \leftarrow \{f_0, \ldots, f_k\}$
initialize: individual $X \leftarrow \{x_0, \ldots, x_m\}; x_i \in F$,
             global solution set $\mathbf{X} \leftarrow \{X\}$,
             subpopulations $\leftarrow \{s_j \subset X\}$,
             non-dominated archive $\mathcal{N} \leftarrow \{\}$
1  while $\mathcal{N}$ has not converged do
2      for each $s_j \in$ subpopulations do
3          $\mathcal{N}''_{s_j} \leftarrow$ MOEA($s$, $n$, $it$, $X$, $F$)
4      end
5      $\mathcal{N}' \leftarrow \{\}$
6      for each variable $x_i$ do
7          $X \leftarrow$ random($\mathbf{X}$)
8          for each $s_j$ where $x_i \in s_j$ do
9              $n'' \leftarrow$ select_best$_{\text{criteria}}$($\mathcal{N}''_{s_j}$)
10             $\mathcal{N}' \leftarrow \mathcal{N}' \cup n''$
11             $X(i) \leftarrow n''_i$
12             $\mathcal{N}' \leftarrow \mathcal{N}' \cup X$
13         end
14     end
15     $\mathcal{N}' \leftarrow$ non-dominated($\mathcal{N}'$)
16     $\mathbf{X} \leftarrow \mathcal{N}'$
17     $\mathcal{N} \leftarrow \mathcal{N} \cup \mathcal{N}'$
18     for each $s_j \in$ subpopulations do
19         $X \leftarrow$ random($\mathbf{X}$)
20         $s_j(X) \leftarrow X$
21         select_worst$_{\text{criteria}}$($\{p_0, \ldots, p_m\} \in s_j$) $\leftarrow X$
22     end
23     $\mathcal{N} \leftarrow$ non-dominated($\mathcal{N}$)
24 end
25 return $\mathcal{N}$
```

V. Experimental Approach

A. Hypothesis

When comparing F-NSGA-II to the other algorithms studied on the Multi-Objective Knapsack problems, we hypothesize the folowing:

1) Using overlapping subpopulations in MOFEA will improve the hypervolume of the approximate Pareto front.
2) Using overlapping subpopulations in MOFEA will improve the spread of the approximate Pareto front.
3) That the non-dominated solution(s) found by MOFEA will contribute a larger percentage of non-dominated solutions to the combined Pareto front of all 5 algorithms' solution sets.

B. Multi-Objective Knapsack Problem

The classic Multi Objective Knapsack problem is defined as follows [30]:

$$\max \sum_{j=1}^{D} a_{ij} x_j, i = 1, 2, \ldots, M$$

$$s.t. \sum_{j=1}^{D} b_{kj} x_j \leq c_k, k = 1, 2, \ldots, C$$

$$x_j \in \{0, 1\}, j = 1, 2, \ldots, D$$

where x is a D-dimensional binary vector, M is the number of objectives, and C is the number of constraints. Specific to this problem, b_{ij} represents the weight of item j inside knapsack i, a_{ij} is the profit of item j inside knapsack i, and c_i is the capacity of knapsack i. This means that the multi-objective part of the problem is defined as having M knapsacks across which D items need to be split according to capacity constraints and value. The constraint value of the weights is set by calculating the total weight for that constraint set and dividing it by 2, as per the original problem proposed by Zitzler and Thiele [7].

We also implemented another type of multi-objective knapsack based on the problem defined by Fortin *et al.* [9]. This problem tries to maximize value, minimize weight, and minimize the difference in weight of the items in a single knapsack, while including constraints placed on the volume and weight of the knapsack using the same method as the multi-knapsack constraint problem. An additional objective to balance the weights is defined as follows:

$$\min \sum_{j,k=1,j\neq k}^{D} |b_j x_j - b_k x_k|.$$

The authors call this type of multi-objective knapsack a balanced knapsack. To add two more objectives, we extended this problem to minimize the overall volume v_j of the items in the knapsack and minimize the difference in volume.

In our experiments, we used a 1000 item knapsack to test the algorithms' performance at a larger scale. For the balanced knapsack, we look at the 3 base objectives (value, weight, and volume) and the extended 5 objective version (balanced weights and volume). The original MO-KS uses fully randomized initialization of the values and weights for all knapsacks and constraints [7]. Therefore, we applied the same approach and initialized the values, weights, and volume randomly as follows: $a_i = [0.1, 100], b_i = [0.1, 5]$, and $v_i = [0.1, 10]$. For the classic knapsack problem, we initialized different sets of values and weights based on the number of objectives and constraints, using the same values as above. For this problem, we considered 3 and 5 objectives with a single weight constraint c_k.

C. Hyperparameter Tuning

We performed a grid search to tune NSGA-II using the following parameter values:

- Algorithm runs: $50, 100, 200, 500$
- Population size: $250, 500, 750, 1000$
- Mutation rate: $0.10, 0.15, 0.20, 0.25$
- Crossover rate: $0.85, 0.90, 0.95, 0.98$

Based on this grid search, we found that running NSGA-II 100 times with a population of 500 was the best combination. For the GA operators, we used tournament selection with $k = 5$ to select the parents, a mutation rate of 0.2 using bitflip mutation and a crossover rate of 0.95 with single-point crossover. The found hyperparameters for NSGA-II were used in the FEA and CCEA implementations of NSGA-II as well. Furthermore, the FEA and CCEA implementations were run for 20 iterations, with two different sizes of subpopulations: 100 and 200, and a 20% overlap for F-NSGA-II, i.e. 20 variables and 40 variables overlap for each subpopulation of size 100 and 200 respectively. The results for each of these runs are discussed.

D. Evaluation Metrics

The hypervolume indicator (HV) is one of the most commonly used evaluation metrics in MOO [31]. Its popularity is partially because the only information needed to calculate the HV of a Pareto Front approximation is a reference point. This is in contrast to measures such as Generational Distance, which requires the true Pareto Front to be known. Since we do not know the true Pareto Front for our problems, the HV is a natural choice to gain insight in the size of the covered objective space [32].

Given k objectives, a set of points $\mathbf{X} \in \mathbb{R}^k$, and a reference point $\mathbf{r} \in \mathbb{R}^k$, the HV of $\mathbf{X}$ is the measure of the region weakly dominated by $\mathbf{X}$ and bound by $\mathbf{r}$ [33]:

$$HV(\mathbf{X}) = \lambda(\{\mathbf{q} \in \mathbb{R}^d | \exists \mathbf{x} \in \mathbf{X} : \mathbf{p} \prec \mathbf{q} \wedge \mathbf{q} \prec \mathbf{r}\}),$$

where λ indicates the Lebesgue measure for k-dimensional set T for $\mathbf{a}, \mathbf{b} \in T$.

$$\lambda(T) = \sum_{i=0}^{k} (b_i - a_i).$$

To assess the diversity of the Pareto Front approximations, we use the spread indicator S [34]:

$$S = \sum_{i=1}^{k} \left[\max_{\mathbf{x} \in \mathbf{X}} \{f_i(\mathbf{x})\} - \min_{\mathbf{x} \in \mathbf{X}} \{f_i(\mathbf{x})\} \right].$$

Thus S corresponds to the sum of the width for each objective, indicating how wide the solutions are spread across the objective space, i.e., a measure of distance instead of volume.

Finally, to compare two Pareto fronts generated by different algorithms directly, we calculate the coverage C of the fronts, denoted as $\mathbf{X}'$ and $\mathbf{X}''$ [35]:

$$C(\mathbf{X}', \mathbf{X}'') = \frac{|\{\mathbf{x}' \in \mathbf{X}' : \exists \mathbf{x}'' \in \mathbf{X}'' : \mathbf{x}'' \preceq \mathbf{x}'\}|}{|\mathbf{X}'|}.$$

This returns a value between 0 and 1, where 0 indicates that no solutions in $\mathbf{X}'$ are dominated by or equal to any solutions in $\mathbf{X}''$, and 1 indicating that all solutions in $\mathbf{X}'$ are dominated by $\mathbf{X}''$. Since the reverse is not a symmetric measure, the metric is calculated and presented for both combinations: $C(\mathbf{X}', \mathbf{X}'')$ and $C(\mathbf{X}'', \mathbf{X}')$. We further adjust this metric to find relative coverage of the non-dominated sets as compared to the total non-dominated set. The total non-dominated set, or union front $\mathbf{X}^*$, is created by combining the results from the set of algorithms g as

$$\mathbf{X}^* = \text{nondom}\left(\bigcup_{i=1}^{g} \mathbf{X}'_i\right).$$

$\mathbf{X}^*$ can then be used to calculate what percentage of each base non-dominated set is included in $\mathbf{X}^*$: $C(\mathbf{X}', \mathbf{X}^*)$. To calculate what percent of $\mathbf{X}^*$ consists of solutions from $\mathbf{X}'_i$, denoted C_p, we adjust the coverage calculation as follows:

$$C_p(\mathbf{X}', \mathbf{X}^*) = \frac{|\{\mathbf{x}' \in \mathbf{X}' : \exists \mathbf{x}^* \in \mathbf{X}^* : \mathbf{x}^* \preceq \mathbf{x}'\}|}{|\mathbf{X}^*|}$$

VI. Results

We ran each algorithm ten times on each of the problem sets and averaged the hypervolume and spread indicator results, as well as the size of the non-dominated population, of those ten runs to get the final metrics shown in Tables I, II and III. Furthermore, we performed an ANOVA test with $\alpha = 5\%$, followed by a paired t-test with $p = 0.05$ to assert statistical significance of the results. CC-NSGA-II results are not found to be significantly different from each other in the majority of cases, with the exception of the 3 objective

TABLE I: Hypervolume results. Underlined results indicate statistically significant results.

		single knapsack		multi knapsack	
	pop.	**3 obj.**	**5 obj.**	**3 obj.**	**5 obj.**
NSGAII	**500**	12.18	12.01	20.99	34.16
CC-NSGAII	**100**	12.24	10.87	17.30	28.76
	200	9.42	10.40	15.99	28.31
F-NSGAII	**100**	12.28	11.92	23.09	39.63
	200	12.22	11.97	22.72	38.56

TABLE II: Spread indicator results. Underlined results indicate statistically significant results.

		balanced knapsack		multi knapsack	
	pop.	**3 obj.**	**5 obj.**	**3 obj.**	**5 obj.**
NSGAII	**500**	38.03	38.24	29.55	32.09
CC-NSGAII	**100**	13.39	8.15	5.89	7.09
	200	5.77	8.58	4.34	6.76
F-NSGAII	**100**	31.34	24.18	15.34	16.91
	200	33.32	30.80	10.39	10.76

TABLE III: Size of the non-dominated solution sets.

		balanced knapsack		multi knapsack	
	pop.	**3 obj.**	**5 obj.**	**3 obj.**	**5 obj.**
NSGA	**500**	166	519	16	38
CC-NSGA	**100**	482	521	21	61
	200	156	476	28	53
F-NSGA	**100**	642	1334	8	14
	200	698	1383	5	9

TABLE IV: Total front coverage results.

		balanced knapsack		multi knapsack	
	pop.	**3 obj.**	**5 obj.**	**3 obj.**	**5 obj.**
NSGAII	**500**	0.64%	14.28%	0.00%	0.00%
CC-NSGAII	**100**	42.69%	16.90%	0.00%	0.00%
	200	0.00%	11.70%	0.00%	0.00%
F-NSGAII	**100**	12.20%	32.85%	100.00%	98.67%
	200	44.47%	24.26%	0.00%	1.13%

single knapsack problem. The opposite is true for F-NSGA-II results, which are significantly different from each other and the results of the other algorithms in most cases. Lastly, fewer statistically significant differences are found between the different hypervolume results, whereas the spread indicator results are largely statistically significant.

To examine the coverage between different solution sets, we randomly pick a single representative run of each algorithm to perform the coverage calculation. To ensure the results are not biased, we perform this process 5 times and average the coverage comparisons for the final results. We present two different coverage results as explained in Section V-D. The total coverage (Table IV) represents the percentage of non-dominated solutions each algorithm contributed to the combined non-dominated solution set. Tables V–VIII show a direct comparison between the algorithms' non-dominated solution sets, where the row algorithm's solution set covers $x\%$ of the column algorithm's solution set.

Finally, we visualize the 3-objective versions of the balanced knapsack and the multi knapsack problems in Figure 2. Each of these figures shows the final non-dominated population of

TABLE V: Single balanced knapsack 3 objectives coverage results

		NSGAII	CC-NSGAII		F-NSGAII	
	pop.	**500**	**100**	**200**	**100**	**200**
NSGAII	**500**	N/A	83.50%	97.21%	28.14%	10.5%
CC-NSGAII	**100**	100.00%	N/A	100.00%	97.85%	98.24%
	200	25.56%	24.92%	N/A	31.55%	5.03%
F-NSGAII	**100**	85.50%	81.47%	96.12%	N/A	35.86%
	200	95.76%	78.27%	99.47%	87.02%	N/A

TABLE VI: Single balanced knapsack 5 objectives coverage results

		NSGAII	CC-NSGAII		F-NSGAII	
	pop.	**500**	**100**	**200**	**100**	**200**
NSGAII	**500**	N/A	99.25%	99.51%	96.45%	76.19%
CC-NSGAII	**100**	76.24%	N/A	99.02%	95.76%	96.90%
	200	64.72%	96.14%	N/A	73.48%	90.41%
F-NSGAII	**100**	65.06%	98.82%	94.69%	N/A	64.42%
	200	56.22%	100.00%	100.00%	73.29%	N/A

TABLE VII: Multi knapsack 3 objectives coverage results

		NSGAII	CC-NSGAII		F-NSGAII	
	pop.	**500**	**100**	**200**	**100**	**200**
NSGAII	**500**	N/A	100.00%	100.00%	0.00%	0.00%
CC-NSGAII	**100**	0.00%	N/A	59.57%	0.00%	0.00%
	200	0.00%	100.00%	N/A	0.00%	0.00%
F-NSGAII	**100**	100.00%	100.00%	100.00%	N/A	100.00%
	200	100.00%	100.00%	100.00%	0.00%	N/A

TABLE VIII: Multi knapsack 5 objectives coverage results

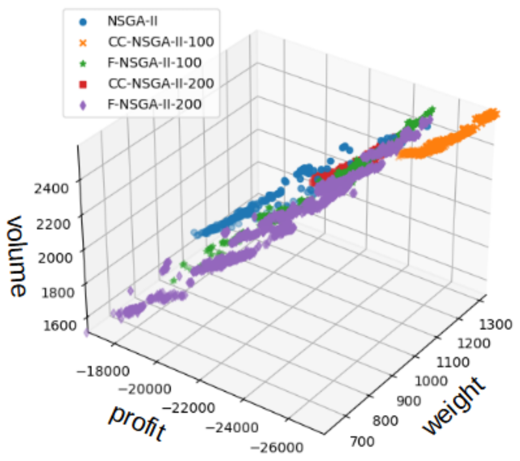
		NSGAII	CC-NSGAII		F-NSGAII	
	pop.	**500**	**100**	**200**	**100**	**200**
NSGAII	**500**	N/A	100.00%	100.00%	0.00%	1.87%
CC-NSGAII	**100**	9.40%	N/A	80%	0.00%	0.00%
	200	0.00%	80.00%	N/A	0.00%	0.00%
F-NSGAII	**100**	100.00%	100.00%	100.00%	N/A	100.00%
	200	100.00%	100.00%	100.00%	2.50%	N/A

a randomly selected run of each of the algorithms, where the x-axis for the single balanced knapsack and the x, y, and z-axes for the multi knapsack show negative values due to the transformation of the knapsack profit maximization to a minimization problem.

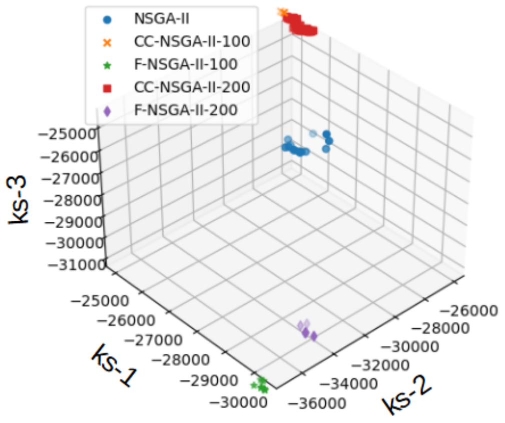
VII. Discussion

Our first hypothesis relating to the hypervolume results can only be confirmed for three of the four problems: the 3-objective balanced knapsack problem and the 3- and 5-objective multi-knapsack problems. No statistically significant differences were found for the 5-objective single knapsack problem. When looking at the spread indicator results, however, we do find statistically significant differences for all four problems. Since single-population NSGA-II has a higher spread indicator for each of them, we cannot confirm our second hypothesis. However, when we look at our coverage results, F-NSGA-II contributes a larger percentage to each problem's combined Pareto front, which confirms our third hypothesis. Based on these results, the spread indicator may have little influence on the quality of the solution set. It is important to note that this is only based on an observed lack of correlation between coverage and spread. These results are by no means conclusive, but warrant further investigation.

A visual inspection of Figure 2a shows that both instances of F-NSGA-II and the 100-population instance of CC-NSGA-II cover more of the space than regular NSGA-II. Taking into



(a) Single balanced knapsack with three objectives.



(b) Classic multi knapsack problem with three objectives.

Fig. 2: Visual representation of the non-dominated population found by each of the algorithms for the three objective versions of the two types of knapsack problems.

consideration that for this problem instance CC-NSGA-II-100 and F-NSGA-II-200 have the highest contribution to the total non-dominated solution set, the visual representation makes sense. CC-NSGA-II-100 is finding solutions in a different part of the space than the other algorithms, but its solutions are not as widely spread across the solutions space, whereas F-NSGA-II has a significantly larger spread than CC-NSGA (Tables I and II), potentially accounting for its large contributions to the total non-dominated solution set. However, NSGA-II has the largest spread indicator while contributing less than 1% of the non-dominated solutions to the total front (Table IV).

Figure 2b shows a different story. With the exception of the two CC-NSGA-II instances, each of the discovered non-dominated solutions is in a different part of the objective space. When looking at Tables IV and VII, the non-dominated solutions discovered by F-NSGA-II with subpopulation size 100 covers all other algorithms' non-dominated solutions. The F-NSGA-II-100 solutions are represented by the green star shaped cluster at the very bottom corner of the image. Since each objective is to be minimized, it makes sense that these solutions, which have converged at the lowest values of the three knapsacks, are dominating the other solutions. This is especially interesting given that the average number of non-dominated solutions found by F-NSGA-II for this problem is smaller than the other algorithms (Table III).

When considering the 5-dimensional problems, F-NSGA-II with subpopulation size 100 once again contributes the largest percentage to the total Pareto front for both problems. Interestingly, NSGA-II's results improve for the 5-objective single knapsack, contrary to the general trend found in the literature, where NSGA-II's performance is often found to deteriorate as objectives increase. When looking at the direct coverage comparison for this problem (Table VI), NSGA-II covers large percentages of the solutions found by the four other algorithms; however, it is only contributing 14.28% to the total front. The larger contribution to the total Pareto front by F-NSGA-II could be explained by the larger number of non-dominated solutions found by the algorithm as compared to NSGA-II (Table III).

The last problem we evaluated was the 5-dimensional multi knapsack. Similar to its 3-objective counter part, the coverage results for this problem are very binary. The F-NSGA-II with a subpopulation size of 100 is the main contributor to the total Pareto front, and when looking at the pairwise comparison, both F-NSGA-II solution sets cover the three other algorithms' solution sets. Another interesting result is that CC-NSGA-II does poorly on both multi-knapsack problems, in that its results are not only 100% covered by F-NSGA-II, but by single population NSGA-II as well.

Overall, both instances of F-NSGA-II do well on all four benchmark problems, indicating that using overlapping subpopulations is beneficial for multi-objective optimization. The use of a smaller subpopulation size in F-NSGA-II seems especially beneficial for finding non-dominated solutions.

VIII. Conclusion

We developed and presented a cooperative co-evolutionary framework for solving multi-objective combinatorial optimization problems that divides the population in subpopulations, which can be distinct or overlapping, and allows usage of any population-based multi-objective algorithm as the base algorithm. We applied our approach to two different implementations of the multi-objective knapsack problem: the multi-knapsack problem, where the number of objectives equals the number of knapsacks [7], and the balanced single knapsack problem [9]. We compared overlapping and distinct subpopulations of different sizes as applied to NSGA-II, as well the single population version of NSGA-II.

Our results indicate that using the MOFEA framework improves NSGA-II's results: F-NSGA-II has the best coverage results on all four solution sets. Furthermore, while the hypervolume results are comparable between the five algorithm implementations, F-NSGA-II does hold a small edge over the other results. On the other hand, the spread found by NSGA-II is generally wider than that found by the other algorithms.

Given the hypervolume and coverage results, this raises the question whether the spread indicator is an appropriate metric for multi-objective optimization.

IX. Future Work

Our future plans include the implementation of different MOEA's in combination with the FEA framework, such as the Strength Pareto Evolutionary Algorithm 2 [36], Non-Dominated Genetic Algorithm III [11], Multi-Objective Evolutionary Algorithm based on Decomposition [37], and the Hypervolume Estimation Algorithm [38]. We also plan to investigate the mechanisms by which using overlapping subpopulations provides benefit in combination with different algorithms, as well as explore how different problems affect the design of the associated factor architectures. This paper only considered static, pre-defined factor architectures. However, the decomposition strategy could influence the results, which is why performing an in-depth study of different decomposition techniques could provide better insight into how the factor architecture used in MOFEA affects MOO problems.

References

[1] A. Blot, M.-E. Kessaci, and L. Jourdan, "Survey and unification of local search techniques in metaheuristics for multi-objective combinatorial optimisation," *Journal of Heuristics*, vol. 24, no. 6, p. 853–877, Dec 2018.

[2] S. Peitz and M. Dellnitz, "A survey of recent trends in multiobjective optimal control—surrogate models, feedback control and objective reduction," *Mathematical and Computational Applications*, vol. 23, no. 2, p. 30, 2018.

[3] R. Marler and J. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, no. 6, pp. 369–395, Apr 2004.

[4] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comp.*, vol. 1, no. 1, pp. 67–82, April 1997.

[5] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization: A short review," in *IEEE Congress on Evolutionary Computation*, 2008, pp. 2419–2426.

[6] S. Strasser, J. Sheppard, N. Fortier, and R. Goodman, "Factored evolutionary algorithms," *IEEE Trans. Evol. Comp.*, vol. 21, no. 2, pp. 281–293, 2017.

[7] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms—a comparative case study," in *Parallel Problem Solving from Nature*. Springer, 1998, pp. 292–301.

[8] R. juan Luo, S. feng Ji, and B. lin Zhu, "A pareto evolutionary algorithm based on incremental learning for a kind of multi-objective multidimensional knapsack problem," *Computers Industrial Engineering*, vol. 135, pp. 537–559, 2019.

[9] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: Evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, jul 2012.

[10] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comp.*, vol. 6, no. 2, pp. 182–197, 2002.

[11] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: Solving problems with box constraints," *IEEE Trans. Evol. Comp.*, vol. 18, no. 4, pp. 577–601, 2013.

[12] M. Carvalho and A. Britto, "Influence of reference points on a many-objective optimization algorithm," in *7th Brazilian Conference on Intelligent Systems (BRACIS)*, 2018, pp. 31–36.

[13] K. Tan, T. Lee, Y. Yang, and D. Liu, "A cooperative coevolutionary algorithm for multiobjective optimization," in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, no. 2, 2004, pp. 1926–1931.

[14] K. Deb, "Multi-objective optimization," in *Search methodologies*. Springer, 2014, pp. 403–449.

[15] C. A. Coello Coello, C. Dhaenens, and L. Jourdan, "Multi-objective combinatorial optimization: Problematic and context," in *Advances in Multi-Objective Nature Inspired Computing*, C. A. Coello Coello, C. Dhaenens, and L. Jourdan, Eds. Springer, 2010, p. 1–21.

[16] M. Ehrgott and X. Gandibleux, "A survey and annotated bibliography of multiobjective combinatorial optimization," *OR-Spektrum*, vol. 22, no. 4, p. 425–460, Nov 2000.

[17] C. Bazgan, H. Hugot, and D. Vanderpooten, "Solving efficiently the 0-1 multi-objective knapsack problem," *Computers & Operations Research*, vol. 36, no. 1, pp. 260–279, 2009.

[18] X. Gandibleux and A. Freville, "Tabu search based procedure for solving the 0-1 multiobjective knapsack problem: The two objectives case," *Journal of Heuristics*, vol. 6, no. 3, pp. 361–383, 2000.

[19] X. Yu, W.-N. Chen, T. Gu, H. Zhang, H. Yuan, S. Kwong, and J. Zhang, "Set-based discrete particle swarm optimization based on decomposition for permutation-based multiobjective combinatorial optimization problems," *IEEE Trans. Cybernetics*, vol. 48, no. 7, pp. 2139–2153, 2017.

[20] S. Verma, M. Pant, and V. Snasel, "A comprehensive review on NSGA-II for multi-objective combinatorial optimization problems," *IEEE Access*, vol. 9, pp. 57 757–57 791, 2021.

[21] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary computation*, vol. 2, no. 3, pp. 221–248, 1994.

[22] M. A. Potter and K. A. D. Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary Computation*, vol. 8, no. 1, pp. 1–29, 2000.

[23] K. Maneeratana, K. Boonlong, and N. Chaiyaratana, "Multi-objective optimisation by co-operative co-evolution," in *Parallel Problem Solving from Nature*. Springer, 2004, p. 772–781.

[24] C. M. Fonseca and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary Computation*, vol. 3, no. 1, pp. 1–16, Mar. 1995.

[25] N. Keerativuttitumrong, N. Chaiyaratana, and V. Varavithya, "Multi-objective co-operative co-evolutionary genetic algorithm," in *Parallel Problem Solving from Nature*. Springer, 2002, pp. 288–297.

[26] D. Zouache, A. Moussaoui, and F. Ben Abdelaziz, "A cooperative swarm intelligence algorithm for multi-objective discrete optimization with application to the knapsack problem," *European Journal of Operational Research*, vol. 264, no. 1, pp. 74–88, 2018.

[27] I. Ben Mansour, M. Basseur, and F. Saubion, "A multi-population algorithm for multi-objective knapsack problem," *Applied Soft Computing*, vol. 70, pp. 814–825, 2018.

[28] H. M. Sahinkoc and Ümit Bilge, "A reference set based many-objective co-evolutionary algorithm with an application to the knapsack problem," *European Journal of Operational Research*, 2021.

[29] N. Fortier, J. Sheppard, and S. Strasser, "Abductive inference in Bayesian networks using distributed overlapping swarm intelligence," *Soft Computing*, vol. 19, no. 4, pp. 981–1001, April 2015.

[30] H. Ishibuchi, N. Akedo, and Y. Nojima, "Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 2, p. 264–283, Apr 2015.

[31] K. Bringmann and T. Friedrich, "Approximation quality of the hypervolume indicator," *Artificial Intelligence*, vol. 195, p. 265–290, Feb 2013.

[32] E. Zitzler, "Evolutionary algorithms for multiobjective optimization: Methods and applications," Ph.D. dissertation, Swiss Federal Institute of Technology, 1999.

[33] D. Brockhoff, T. Friedrich, and F. Neumann, "Analyzing hypervolume indicator based algorithms," in *Parallel Problem Solving from Nature*. Springer, 2008, p. 651–660.

[34] H. Ishibuchi and Y. Shibata, "Mating scheme for controlling the diversity-convergence balance for multiobjective optimization," in *ACM Genetic and Evolutionary Computation Conference*, 2004, p. 1259–1271.

[35] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, p. 173–195, Jun 2000.

[36] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm," *TIK-report*, vol. 103, 2001.

[37] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.

[38] J. Bader and E. Zitzler, "Hype: An algorithm for fast hypervolume-based many-objective optimization," *Evolutionary computation*, vol. 19, no. 1, pp. 45–76, 2011.