Multi-Objective Evolutionary Approaches for the Knapsack Problem with Stochastic Profits

Kokila Kasuni Perera[0000−0002−0748−0815], Frank Neumann[0000−0002−2721−3618], and Aneta Neumann[0000−0002−0036−4782]

Optimisation and Logistics,
School of Computer and Mathematical Sciences,
The University of Adelaide, Adelaide, Australia

Abstract. Uncertainties in real-world problems impose a challenge in finding reliable solutions. If mishandled, they can lead to suboptimal or infeasible solutions. Chance constraints are a natural way to capture uncertain problem parameters. They model probabilistic constraints involving the stochastic parameters and an upper bound of probability that mimics the confidence level of the solution. We focus on the knapsack problem with stochastic profits to guarantee a certain level of confidence in the profit of the solutions. We present a bi-objective fitness formulation that uses expected profit and standard deviation to capture the chance constraints. This formulation enables optimising the problem independent of a specific confidence level. We evaluate the proposed fitness formulation using well-known evolutionary algorithms GSEMO, NSGA-II and MOEA/D. Moreover, we introduce a filtering method that refines the interim populations based on the confidence levels of its solutions. We evaluate this method by applying it along with GSEMO to improve the quality of its population during optimisation. We conduct extensive experiments to show the effectiveness of these approaches using several benchmarks and present a detailed analysis of the results.

Keywords: Evolutionary algorithms · Multi-objective optimisation · Chance constraints

1 Introduction

Real-world optimisation problems become challenging when they have uncertain problem parameters or uncontrollable environmental changes. In most real-world situations, these uncertainties are inevitable, such as highly dynamic load demands in the power grids [5], uncertain cost estimates in budgeting [4], uncertain weather conditions affecting transportation systems [17], and mine optimisation under uncertainty [31,36]. It is crucial to capture the effects of uncertain parameters to identify risks and avoid disastrous failures and high recovery costs. For example, weather conditions heavily impact the transportation time schedules of commercial vessels. In this scenario, uncertainties in transportation time impact predetermined deadlines and could result in additional costs for maritime companies [17]. It is important to consider the implications of stochastic problem components to have a more realistic view of the problem conditions and identify safe and reliable solutions.

Chance constraints are a natural way to model uncertainties in problems. A chance constraint defines a small valued upper bound for the probability ($0 < \alpha < 1/2$) that a particular constraint may be violated [10,25,9]. It means a solution is feasible if the likelihood of its constraint violation is equal to or below α. Moreover, this probabilistic constraint allows us to attribute a certain confidence level to the solutions of a stochastic optimisation problem. Chance constraints can be used to design practical applications for stochastic optimisation problems in many fields such as mining [40,31], power systems [13,5], communication systems [1] and transportation [37,17].

In this paper, we consider a variation of the classical knapsack problem. The deterministic knapsack problem [16] is a classical NP-hard combinatorial optimisation problem. It has been considered in different settings in the literature, such as with stochastic weights [38,39], with stochastic profits [24] and other dynamic and stochastic settings [12,3,2,33,34,32,34]. This paper focuses on the knapsack variant with deterministic weights and stochastic profits. While the deterministic constraint on the weights remains the same as in the classical knapsack problem, we introduce a chance constraint on the profits to capture the uncertainties in profits [24]. This problem model can be beneficial in many real-world problems, such as modelling complex planning problems like mine design and mining activity scheduling, which will allow discovering plans to achieve the maximum profit with a higher confidence level.

We consider the evolutionary algorithms to address the target problem. Evolutionary approaches perform well in addressing stochastic optimisation problems, including chance-constrained problems [3,10,30]. In particular, multi-objective evolutionary approaches optimise the problem by considering multiple objectives simultaneously. They generate a set of solutions that gives a trade-off of optimal solutions concerning the objectives. This final population provide more insights into improving the algorithms and search space than having a single solution as the outcome [6,7]. Therefore, multi-objective algorithms help one to make informed decisions when selecting a solution to implement.

1.1 Related Work

The early literature on evolutionary computation for chance-constrained problems considers computationally expensive methods like simulations and sampling to cater for chance constraints [11,2,42,15,19,18,20,9]. More recent studies have looked into tail-bound inequalities, which more efficiently deal with chance constraints [24,38,3,43,29,28,23,22].

The chance-constrained knapsack problem with deterministic profits and stochastic weights are considered in several papers [38,39,28]. Yue et al. 2019 [38] present how to use well-known deviation inequalities like Chebyshev's inequality and Chernoff bound to estimate the probability of constraint violation.

In [39], where the same knapsack problem variation is considered, they introduce problem-specific operators for EAs with both single- and multi-objective formulations. Assmi et al. [3] study the evolutionary approaches focusing on the dynamic chance-constrained knapsack problem with stochastic weights and a dynamic weight bound. In addition to the objective function on the profit of a given stochastic solution, a second objective is introduced to address the dynamic capacity constraint. It captures the minimal capacity bound for the solution that meets the chance constraints.

Run-time analysis is an essential topic in studying problems with chance constraints. The first paper on run time analysis for chance constraint problems considers the knapsack problem with stochastic weights [26]. This work considers different cases of the problem and studies the run time of (1+1) EA for them. In [41], they perform the run time analysis of simple EAs for chance-constrained knapsack problems with uniform weights. The papers [27] and [35] study the run time of simple EAs for different chance-constrained problems. In[27], the authors consider single- and multi-objective EAs for chance-constrained problems with customarily distributed random problem variables. They also show how to use the proposed evolutionary approaches for chance-constrained minimum spanning tree problems [27]. In [35], they analyse the run time of random local search and (1+1) EA for the chance-constrained makespan problem.

In the study [24], the authors study single objective optimisation of profit chance-constrained knapsack problem with simple evolutionary algorithms. Those algorithms include (1+1) EA with standard bit-flip and heavy-tail mutation operators and population-based (μ+1) EA with a specific crossover operator specific for the knapsack problem. This study evaluates the performance of all these algorithms using the single objective fitness evaluation. The overall results show that (1+1) EA with heavy tail mutation operator significantly improved over other algorithms.

1.2 Our Contribution

We introduce a bi-objective fitness function motivated by the recent study on the evolutionary optimisation of chance-constrained problems by computing the trade-offs concerning the expected value and variance of solutions presented in [27]. The significance of this function is that it evaluates the fitness of a solution independent of a specific confidence level in profit (i.e. a specific value of α). Since this generates a set of solutions that gives a trade-off of the objectives, it allows one to make more informed decisions when selecting a solution to implement. For example, to identify the solution that gives the best profit with a particular α value, we can calculate the profit of all the solutions for that confidence level and select the solution that gives the best profit among the final population. Also, deciding on the confidence levels before running optimisation algorithms is not required when using the introduced fitness formulation.

This paper introduces a filtering method for chance-constrained optimisation as a key algorithmic contribution. This method improves the effects of the proposed fitness function. We consider this filtering method with the GSEMO as a separate algorithm to solve the chance-constrained problem.

We evaluate the effectiveness of the fitness formulation with the proposed algorithm and three well-known multi-objective evolutionary algorithms. We consider the global simple evolutionary multi-objective optimiser (GSEMO) [14] and state-of-the-art algorithms: non-dominated sorting genetic algorithm (NSGA-II) [8] and multi-objective evolutionary algorithm based on decomposition (MOEA/D) [44]. Each of these algorithms implements a unique evolutionary approach and allows us to see the effectiveness of the proposed fitness function under different evolutionary techniques.

The rest of this paper is structured as follows. Section 2 covers the preliminaries, including the formal introduction of the problem and profit estimates based on concentration bounds. In Section 3, we discuss the multi-objective formulation, including the fitness function and how to use the probability bounds to estimate the confidence in the solutions' profit. Afterwards, we introduce the algorithms and how they address the chance-constrained problem in Section 4. Finally, we present the experimental settings and detailed analysis of the results in Section 5, followed by the conclusions of this work in Section 6.

2 Preliminaries

The classical knapsack problem can be defined as follows. The input is given as n elements $1, \ldots, n$ with associated profits p_i and weight w_i, $1 \leq i \leq n$, and weight bound B. A possible solution $x \in \{0,1\}^n$ is represented as a bitstring of length n such that $x_i = 1$ holds *iff* the element i is selected in x. Given the profit of x as $p(x) = \sum_{i=1}^{n} p_i x_i$ and weight $w(x) = \sum_{i=1}^{n} w_i x_i$, the goal in the classical knapsack problem is to find the solution x^* that maximises $p(x)$ subject to the weight constraint $w(x) \leq B$, i.e. $x^* = \arg\max_{x \in \{0,1\}^n} \{p(x) \mid w(x) \leq B\}$ holds.

When the profits p_i of the knapsack elements are stochastic, the profit of a solution is uncertain and varies from the expected profit. We use a chance constraint on the profit to capture this stochastic behaviour. This constraint ensures that for each feasible solution x, the probability that the profit will drop below the maximal profit (P) is at most a small probability $0 < \alpha < 1/2$.

We can formally present this problem as follows:

$$\max P \tag{1}$$

$$\text{subject to } Pr(p(x) < P) \leq \alpha \tag{2}$$

$$\text{and} \qquad w(x) \leq B \tag{3}$$

When the profits p_i of knapsack elements are stochastic, it is not always possible to calculate the exact maximal profit of a solution for which it meets the chance constraint given by Equation 2. Such calculations are often only possible for special cases, like where the profits are independent and normally distributed random variables. The most common alternative in literature is to consider the tail-bound inequalities, which apply under different conditions of profit distribution. They can be used to define an upper bound on the probability of constraint violation [21] and to formulate the maximal profit of a solution subject to the chance constraint. In recent literature, Chebyshev inequality and Hoeffding bound have been considered to derive profit estimates for the knapsack problem with stochastic profits [24]. We present below the profit estimates $\hat{P}_{\text{Cheb}}(x, \alpha)$ and $\hat{P}_{\text{Hoef}}(x, \alpha)$ from the literature [24], that evaluate the maximal profit (P) of a feasible solution x subject to the violation of profit constraint given by Equation (2) is at most a given α.

Let $\mu(x)$ and $v(x)$ be the expectation and variance of profit of x, then based on Chebyshev inequality [21], $\hat{P}_{\text{Cheb}}(x, \alpha)$ gives an estimate for the profit of solution x with α as the maximum chance of constraint violation as follows:

$$\hat{P}_{\text{Cheb}}(x, \alpha) = \mu(x) - \sqrt{(1-\alpha)/\alpha} \cdot \sqrt{v(x)}. \tag{4}$$

The Hoeffding bound [21] can be used to formulate a profit estimate if the profits are independent and distributed uniformly with the same dispersion. Let the profits of elements be uniformly distributed as $p_i \in [\mu_i - \delta, \mu_i + \delta]$ where μ_i is the expected profit of element i and δ is the dispersion of profits. Then, we can estimate the profit of x holding the chance constraint with a given α as follows:

$$\hat{P}_{\text{Hoef}}(x, \alpha) = \mu(x) - \delta \cdot \sqrt{\ln(1/\alpha) \cdot 2 \cdot |x|_1}. \tag{5}$$

3 Methods

One of the main contributions of this work is the introduction of multi-objective fitness formulations for the profit chance-constrained problem. Here, we introduce the bi-objective fitness function for the problem and how the confidence levels are associated with the solutions in a population optimised using this function.

3.1 Fitness Formulations

Now, we introduce the fitness formulation as $g(x) = (\mu(x), v(x))$, which considers the two objectives based on the expected value and variance of the solution's profit. Given that $v_{\max} = \sum_{i=1}^{n} \sigma_i^2$ denotes the maximal variance of the objectives, the objectives $\mu(x)$ and $v(x)$ are defined as,

$$\mu(x) = \begin{cases} \sum_{i=1}^{n} \mu_i x_i & w(x) \leq B \\ B - w(x) & \text{otherwise} \end{cases} \tag{6}$$

$$v(x) = \begin{cases} \sum_{i=1}^{n} \sigma_i^2 x_i & w(x) \leq B \\ v_{\max} + (w(x) - B) & \text{otherwise} \end{cases} \tag{7}$$

These two conflicting objectives reflect the requirement of maximising profit while minimising the chances of uncertainties. Therefore, we maximise the objective corresponding to the expected profit ($\mu(x)$) and minimise the objective corresponding to the variance of profits ($v(x)$). Given two feasible solutions x and

y, we say that solution x dominates y $(x \succeq y)$ *iff* $\mu(x) \geq \mu(y) \wedge v(x) \leq v(y)$ and we say that x strongly dominates y $(x \succ y)$ *iff* $x \succeq y \wedge \mu(x) > \mu(y) \wedge v(x) < v(y)$. Furthermore, if a solution exceeds the weight bound $(w(x) > B)$, the value of each objective is penalised accordingly to capture the deterministic constraint on weights. This ensures that any feasible solution dominates the infeasible solution instances.

3.2 Identifying the Best Solution

The fitness function g allows bi-objective optimisation of the problem independent of a specific confidence level (α) in the profit. The outcome of an evolutionary algorithm that uses g to evaluate the fitness of solutions is a set of solutions giving a trade-off of the objectives μ and v. Given a particular confidence level α, we need to calculate the profits of all solutions in the population for that confidence level and choose the one giving the highest profit value as the best solution in the population for the given α. We need to use the profit estimates $\hat{P}_{\text{Cheb}}$ and $\hat{P}_{\text{Hoef}}$ (given in Equation 4 and 5) to calculate the profit of each solution for the confidence level α. We define the best solution x^* that gives the highest profit value for α, as $\arg\max_{x \in P} \hat{P}_{\text{Cheb}}(x, \alpha)$ or $\arg\max_{x \in P} \hat{P}_{\text{Hoef}}(x, \alpha)$.

3.3 Level of Confidence in a Solution's Profit

The fitness function g allows multi-objective optimisers to generate a population, giving different solutions that maximise profit for specific confidence levels. Given a population, we can associate a confidence interval for each solution, such that the solution shows the best profit in the population for any confidence level in the interval. First, we need to calculate the confidence level threshold for a pair of solutions, that one solution gives a better profit than the other. Here, we consider two solutions that do not dominate each other, which means each solution is better concerning at least one objective, μ or v. Therefore, we consider non-dominating solution pairs when introducing Theorem 1 and 2.

We obtain the first theorem concerning $\hat{P}_{\text{Cheb}}$ profit estimate (Equation 4), which allows us to define a minimum confidence level that the profit of one solution becomes better than another given solution.

Theorem 1. *Let* $0 < \alpha < 1,$ *and* x *and* y *be two feasible solutions that satisfy* $\mu(x) > \mu(y)$ *and* $v(x) > v(y)$. *If* $\alpha \geq \frac{1}{1+(R_{Cheb}(x,y))^2}$ *holds such that* $R_{Cheb}(x, y) = \frac{\mu(x)-\mu(y)}{\sqrt{v(x)}-\sqrt{v(y)}}$ *then* $\hat{P}_{Cheb}(x, \alpha) \geq \hat{P}_{Cheb}(y, \alpha)$.

Proof. We have,

$$\alpha \geq 1/\left(1 + R_{\text{Cheb}}(x, y)^2\right)$$
$$\iff \quad R_{\text{Cheb}}(x, y)^2 \geq (1 - \alpha)/\alpha$$

As we assume $0 < \alpha < 1$, $\mu(x) > \mu(y)$ and $v(x) > v(y)$, we have $R_{\text{Cheb}}(x,y) > 0$ and $(1-\alpha)/\alpha > 0$. This implies,

$$\begin{aligned}
& & R_{\text{Cheb}}(x,y) \geq \sqrt{(1-\alpha)/\alpha} \\
&\Longleftrightarrow & \frac{\mu(x)-\mu(y)}{\sqrt{v(x)}-\sqrt{v(y)}} \geq \sqrt{(1-\alpha)/\alpha} \\
&\Longleftrightarrow & \mu(x)-\sqrt{(1-\alpha)/\alpha}\cdot\sqrt{v(x)} \geq \mu(y)-\sqrt{(1-\alpha)/\alpha}\cdot\sqrt{v(y)} \\
&\Longleftrightarrow & \hat{P}_{\text{Cheb}}(x,\alpha) \geq \hat{P}_{\text{Cheb}}(y,\alpha)
\end{aligned}$$

This completes the proof. □

The above theorem defines a threshold α value for the profit between two solutions. Let the two feasible solutions x and y satisfy $\mu(x) > \mu(y)$ and $v(x) > v(y)$, we define $\alpha^*_{\text{Cheb}}(x,y) = 1/\left(1+R_{\text{Cheb}}(x,y)^2\right)$, that gives

$$\begin{aligned}
\hat{P}_{\text{Cheb}}(x,\alpha) = \hat{P}_{\text{Cheb}}(y,\alpha) \textit{ iff } \alpha = \alpha^*_{\text{Cheb}}(x,y) \\
\hat{P}_{\text{Cheb}}(x,\alpha) > \hat{P}_{\text{Cheb}}(y,\alpha) \textit{ iff } \alpha > \alpha^*_{\text{Cheb}}(x,y) \\
\hat{P}_{\text{Cheb}}(x,\alpha) < \hat{P}_{\text{Cheb}}(y,\alpha) \textit{ iff } \alpha < \alpha^*_{\text{Cheb}}(x,y)
\end{aligned}$$

Next, we present a theorem based on $\hat{P}_{\text{Hoef}}$ (Equation 5), which defines a minimum confidence level for which one solution in a pair of non-dominated solutions gives a better profit estimate. $\hat{P}_{\text{Hoef}}$ is applicable when the profits have the same dispersion, and the variances depend on the number of elements in the solution. Similarly, the conditions in Theorem 2 are defined based on the expected profit and number of items.

Theorem 2. *Let $0 < \alpha < 1$, and x and y be two feasible solutions such that $\mu(x) > \mu(y)$ and $|x|_1 > |y|_1$, holds. If $\alpha \geq e^{-R_{Hoef}(x,y)^2}$ holds such that $R_{Hoef}(x,y) = \frac{\mu(x)-\mu(y)}{\delta\left(\sqrt{2|x|_1}-\sqrt{2|y|_1}\right)}$ then $\hat{P}_{Hoef}(x,\alpha) \geq \hat{P}_{Hoef}(y,\alpha)$.*

Proof. We have,

$$\begin{aligned}
& & \alpha \geq e^{-R_{\text{Hoef}}(x,y)^2} \\
&\Longleftrightarrow & R_{\text{Hoef}}(x,y)^2 \geq \ln(1/\alpha)
\end{aligned}$$

As we assume $0 < \alpha < 1$, $\mu(x) > \mu(y)$, $|x|_1 > |y|_1$ and $\delta > 0$, we have $R_{\text{Hoef}}(x,y) > 0$ and $\ln\frac{1}{\alpha} > 0$. This implies,

$$\begin{aligned}
& & R_{\text{Hoef}}(x,y) \geq \sqrt{\ln(1/\alpha)} \\
&\Longleftrightarrow & \frac{\mu(x)-\mu(y)}{\delta\left(\sqrt{2|x|_1}-\sqrt{2|y|_1}\right)} \geq \sqrt{\ln(1/\alpha)} \\
&\Longleftrightarrow & \mu(x)-\delta\sqrt{\ln(1/\alpha)\cdot 2|x|_1} \geq \mu(y)-\delta\sqrt{\ln(1/\alpha)\cdot 2|y|_1} \\
&\Longleftrightarrow & \hat{P}_{\text{Hoef}}(x,\alpha) \geq \hat{P}_{\text{Hoef}}(y,\alpha)
\end{aligned}$$

This completes the proof. □

Given two feasible solutions x and y that satisfy $\mu(x) > \mu(y)$ and $|x|_1 > |y|_1$, we define a threshold value, $\alpha^*_{\text{Hoef}}(x, y) = e^{-R_{\text{Hoef}}(x,y)^2}$ that gives,

$$\hat{P}_{\text{Hoef}}(x, \alpha) = \hat{P}_{\text{Hoef}}(y, \alpha) \textit{ iff } \alpha = \alpha^*_{\text{Hoef}}(x, y)$$
$$\hat{P}_{\text{Hoef}}(x, \alpha) > \hat{P}_{\text{Hoef}}(y, \alpha) \textit{ iff } \alpha > \alpha^*_{\text{Hoef}}(x, y)$$
$$\hat{P}_{\text{Hoef}}(x, \alpha) < \hat{P}_{\text{Hoef}}(y, \alpha) \textit{ iff } \alpha < \alpha^*_{\text{Hoef}}(x, y).$$

It is important to note that these thresholds $\alpha^*_{\text{Cheb}}(x, y)$ and $\alpha^*_{\text{Hoef}}(x, y)$ are applicable under the same conditions that apply to the corresponding to the profit estimates.

The thresholds of confidence levels for solution pairs allow us to derive the confidence value interval for which a particular solution gives the best profit value over other solutions in a population. We define this confidence value range for solutions in a population optimised using the fitness function g. First, we define an ordering of the solutions for such a population.

Let the final population has m solutions $\{x^1, \ldots x^m\}$ sorted by expected profit such that $\mu(x^1) \geq \mu(x^2) \geq \ldots \geq \mu(x^m)$. Since it uses the fitness function g for optimisation, the above ordering of the solutions satisfies $v(x^1) \geq v(x^2) \geq \ldots \geq v(x^m)$. Furthermore, if the knapsack elements have the same dispersion, this ordering holds $|x^1|_1 \geq |x^2|_1 \geq \ldots \geq |x^m|_1$.

For all solutions pairs $x^i, x^j \in \{1, \ldots, m\}$, we calculate the confidence level using either $\alpha^*_{\text{Cheb}}(x^i, x^j)$ or $\alpha^*_{\text{Hoef}}(x^i, x^j)$ and for $i = 0$ and $j = 0$ we consider the confidence level as 1. Then, the confidence interval for the profit of solution x^k is given by the following intervals,

$$\max_{i=k+1}^{m} \alpha^*_{\text{Cheb}}(x^i, x^k) \leq \alpha \leq \min_{i=0}^{k-1} \alpha^*_{\text{Cheb}}(x^i, x^k) \tag{8}$$

$$\max_{i=k+1}^{m} \alpha^*_{\text{Hoef}}(x^i, x^k) \leq \alpha \leq \min_{i=0}^{k-1} \alpha^*_{\text{Hoef}}(x^i, x^k) \tag{9}$$

Notably, only some of the solutions in the final population will have a non-empty interval. A solution with an empty α interval implies that it does not give the best profit value for any confidence level.

4 Algorithms for the Chance Constrained Knapsack Problem

This study considers three widely used multi-objective evolutionary algorithms: GSEMO, NSGA-II and MOEA/D. GSEMO is the simplest form of a multi-objective evolutionary algorithm. It uses dominance between solutions to select the population for the next generation. The specific steps of GSEMO are given in Algorithm 1. The initial population contains a solution where each bit is randomly chosen. In each iteration, a parent solution x is randomly selected from the population S, and an offspring solution y is generated by flipping each bit in x with a probability of $1/n$. If the existing solutions do not dominate y,

Algorithm 1 GSEMO

```
1: Choose $x \in \{0,1\}^n$ uniformly at random ;
2: $S \leftarrow \{x\}$;
3: while stopping criterion not met do
4:     choose $x \in S$ uniformly at random;
5:     $y \leftarrow$ flip each bit of $x$ independently with probability of $\frac{1}{n}$;
6:     if $(\nexists w \in S : w \succ y)$ then
7:         $S \leftarrow (S \cup \{y\}) \setminus \{z \in S \mid y \succeq z\}$ ;
8:     end if
9: end while
```

Algorithm 2 Filtering Method

```
0: Input: Population $P_0 = x^1, ..., x^m$ ordered by the decreasing order of $\mu(x^i)$;
1: for $k = 1$ to $m$ do
2:     Calculate $\alpha^*(x^i, x^k)$ for $i \in i, \ldots, m$ using either $\alpha^*_{\text{Cheb}}$ or $\alpha^*_{\text{Hoef}}$
3:     if $\min_{i=0}^{k-1} \alpha^*(x^i, x^k) \geq \max_{j=k+1}^{m} \alpha^*(x^j, x^k)$ then
4:         $P_1 \cup \{x^k\}$
5:     end if
6: end for
7: return $P_1$
```

it is added to S, and all solutions dominated by y are removed from S. These iterations repeat until the given number of fitness evaluations are completed.

The final population of GSEMO contains solutions that do not give the best profit value for any probability value for α. Such solutions do not add value to the optimisation goal of finding the best solutions with given confidence levels. For instance, Figure 1 presents final populations from GSEMO using 10 million fitness evaluations on two problem instances that are used in the experiments. Only the solutions marked by a blue star have a valid confidence level interval, and those intervals compose the complete probability range [0,1]. Only these solutions in the final population will be useful at the end of the optimisation.

We introduce a filtering method to remove the solutions without a valid α interval from the interim populations. It is regularly applied to the interim populations of GSEMO after a certain amount of fitness evaluations. Here, the filtering method uses Equation 8 or 9 to calculate the α intervals of the solutions. As the filtering method keeps only the solutions with valid α intervals, it increases the chances that the new solutions are improved upon these solutions and eventually improves the quality of the final population.

The steps of the filtering method are given in Algorithm 2. It takes the population P_0 as the input, which can be either the final population or an interim population created during the execution of GSEMO. Population P_0 needs solutions in the decreasing order of the expected profit value. For each solution x^k, we consider its the confidence interval (Equation 8 and 9) and add x^k to the new population P_1 *iff* this interval is non-empty.

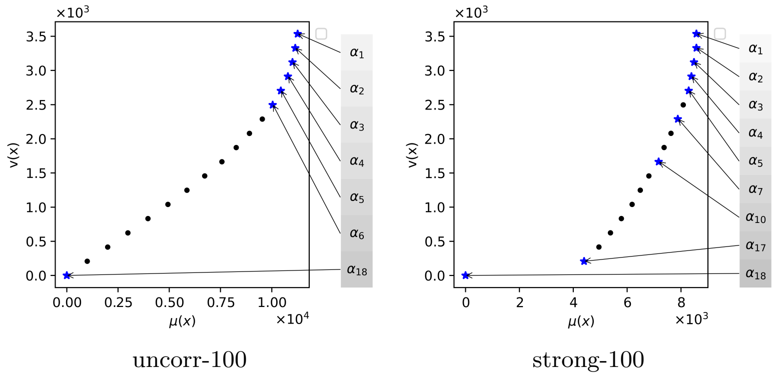

Fig. 1. Trade-offs of objectives in the final populations from GSEMO.

NSGA-II is a prominent multi-objective evolutionary algorithm focusing on diversity by finding near-optimal solutions [8]. If we consider one iteration of the evolutionary process of NSGA-II, it creates an offspring population with an equal number of solutions as the parent population. NSGA-II generate offspring solutions by selecting two solutions using binary tournament selection and then applying uniform crossover and bit-flip mutation (with a probability of $1/n$). Next, parent and offspring populations are considered together and divided into non-dominating solution fronts. The solutions for the new population are selected from the lowest to higher-ranking fronts. If the new population does not have space to fit a front completely, the remainder of the solutions are selected considering the crowding distance. Considering crowding distance allows diverse solutions to be selected for the population. The NSGA-II stops when the required number of fitness evaluations are completed.

MOEA/D optimises a multi-objective problem by decomposing it into multiple single-objective subproblems and solving them collaboratively. We set the population size m equal to the number of knapsack elements ($m = n$). Then, we define a set of even spread weight vectors $\lambda^1 \dots \lambda^m$. Each weight vector is of length two, representing the decomposition of the problem into the two subproblems concerning two objectives. The distance between the weight vectors determines the neighbourhood of a solution. We use the following aggregation function to evaluate solutions against each weight vector λ^j and reference point z,

$$g^{te}(x|\lambda^j, z) = \max\{\lambda_1^j|\mu(x) - z_1|, \lambda_2^j|v(x) - z_2|\}$$

The reference point z is determined by the best objective values (μ and v) given by different solutions in the current population. This point helps to guide the optimisation process [44]. We maintain configurations for MOEA/D similar to other algorithms we study in this paper. We initialise the population with randomly generated solutions. To generate offspring solutions, we use uniform crossover and bit-flip mutation (with a probability of $1/n$). Furthermore, we run MOEA/D until it completes the given number of fitness evaluations.

Table 1. Experimental Results using Chebyshev's inequality

		GSEMO (1)			GSEMO+Filtering (2)			NSGA-II (3)			MOEA/D (4)		
α	δ	mean	std	stat	mean	std	stat	mean	std	stat	mean	std	stat
uncorr-100 (B = 2407)													
0.1	25	11029.6826	76.32	$2^-3^+4^+$	**11085.6072**	0.00	$1^+3^+4^+$	11030.2076	66.36	1^-2^-	10989.8740	101.49	1^-2^-
	50	10862.4750	61.41	3^+4^+	**10907.0000**	0.00	3^+4^+	10833.2699	64.13	1^-2^-	10708.3350	137.02	1^-2^-
0.01	25	10620.5264	71.04	$2^-3^+4^+$	**10672.3379**	0.00	$1^+3^+4^+$	10621.9585	60.35	1^-2^-	10583.7093	94.69	1^-2^-
	50	10044.4941	49.77	3^+4^+	**10079.9649**	0.00	3^+4^+	10020.0651	53.33	1^-2^-	9910.1844	124.55	1^-2^-
0.001	25	9345.9245	55.26	$2^-3^+4^+$	**9384.5150**	0.00	$1^+3^+4^+$	9349.7796	42.03	1^-2^-	9318.3976	74.42	1^-2^-
	50	7495.5153	17.96	3^+4^+	**7502.7716**	0.00	3^+4^+	7488.2131	23.20	1^-2^-	7429.7797	83.79	1^-2^-
strong-100 (B = 4187)													
0.1	25	8507.0584	130.14	2^-4^+	**8606.3413**	87.30	1^+4^+	8582.6920	85.10	4^+	8402.1225	91.54	$1^-2^-3^-$
	50	8368.0306	94.01	4^+	**8422.6322**	67.23	4^+	8379.6138	85.94	4^+	8012.8210	220.02	$1^-2^-3^-$
0.01	25	8083.9678	106.20	2^-4^+	**8170.3360**	69.58	1^+4^+	8144.0469	68.35	4^+	7991.9311	75.75	$1^-2^-3^-$
	50	7502.9361	59.85	2^-4^+	**7549.4937**	41.27	$1^+3^+4^+$	7510.8080	54.00	2^-4^+	7221.6377	175.65	$1^-2^-3^-$
0.001	25	6770.3193	41.13	2^-4^+	**6814.1197**	20.77	$1^+3^+4^+$	6788.1211	28.24	2^-4^+	6720.6049	47.05	$1^-2^-3^-$
	50	**4957.5449**	36.78	2^+4^+	4894.0039	60.21	1^-3^-	4945.4605	47.34	2^+4^+	4847.9857	65.58	1^-3^-
uncorr-300 (B = 6853)													
0.1	25	33935.4067	205.62	2^-	**34286.1802**	147.53	$1^+3^+4^+$	33998.1173	348.47	2^-	33804.0944	354.70	2^-
	50	33571.9980	260.86	2^-4^+	**33967.3813**	159.24	$1^+3^+4^+$	33713.5615	226.47	2^-4^+	32117.5366	642.48	$1^-2^-3^-$
0.01	25	33237.8865	200.46	2^-	**33577.9421**	141.75	$1^+3^+4^+$	33295.2934	339.93	2^-	33108.5096	343.50	2^-
	50	32180.0106	245.88	2^-4^+	**32551.5342**	144.90	$1^+3^+4^+$	32311.5529	213.24	2^-4^+	30782.0871	614.16	$1^-2^-3^-$
0.001	25	31066.6084	186.36	2^-	**31372.5619**	122.86	$1^+3^+4^+$	31106.1048	313.81	2^-	30942.2692	309.35	2^-
	50	27843.2948	203.73	2^-4^+	**28141.7188**	105.94	$1^+3^+4^+$	27950.6523	176.37	2^-4^+	26624.6825	533.58	$1^-2^-3^-$
strong-300 (B = 13821)													
0.1	25	23809.6581	433.25	2^-3^-	**24369.6211**	216.96	1^+4^+	24325.6409	294.11	1^+4^+	23330.6120	412.53	2^-3^-
	50	23594.2993	335.65	$2^-3^-4^+$	**24135.2769**	220.45	1^+4^+	24003.0894	230.30	1^+4^+	22314.8864	685.49	$1^-2^-3^-$
0.01	25	23176.9548	406.27	$2^-3^-4^+$	**23703.0401**	197.34	1^+4^+	23648.8178	262.07	1^+4^+	22734.4253	387.17	$1^-2^-3^-$
	50	22322.7651	282.01	$2^-3^-4^+$	**22797.4912**	177.42	1^+4^+	22656.7197	185.32	1^+4^+	21176.4653	610.85	$1^-2^-3^-$
0.001	25	21208.9163	322.79	$2^-3^-4^+$	**21626.7053**	138.64	1^+4^+	21539.7126	166.67	1^+4^+	20879.1124	311.52	$1^-2^-3^-$
	50	18388.5805	159.45	2^-4^+	**18647.0894**	70.19	$1^+3^+4^+$	18495.6543	100.69	2^-4^+	17655.0770	402.61	$1^-2^-3^-$
uncorr-500 (B = 11243)													
0.1	25	57076.8361	748.93	2^-3^-	**58431.4168**	311.58	$1^+3^+4^+$	57912.4733	648.06	$1^+2^-4^+$	57004.9636	781.84	2^-3^-
	50	56690.8982	859.24	$2^-3^-4^+$	**58120.8249**	314.31	1^+4^+	57547.3013	565.83	1^+4^+	52794.2727	1321.93	$1^-2^-3^-$
0.01	25	56197.0249	738.34	2^-3^-	**57528.4355**	304.43	$1^+3^+4^+$	57017.0360	635.76	$1^+2^-4^+$	56127.1101	769.25	2^-3^-
	50	54931.1821	829.59	$2^-3^-4^+$	**56312.8274**	298.86	1^+4^+	55758.1900	544.95	1^+4^+	51138.2861	1284.89	$1^-2^-3^-$
0.001	25	53456.0312	705.50	2^-3^-	**54715.2628**	282.53	$1^+3^+4^+$	54227.7939	597.99	$1^+2^-4^+$	53392.0959	729.83	2^-3^-
	50	49451.2762	737.56	$2^-3^-4^+$	**50683.8705**	252.35	$1^+3^+4^+$	50185.9102	481.83	$1^+2^-4^+$	45981.8039	1169.98	$1^-2^-3^-$
strong-500 (B = 22223)													
0.1	25	38822.1695	692.12	2^-3^-	**40391.0362**	449.82	1^+4^+	40108.0207	492.45	1^+4^+	38258.0920	779.48	2^-3^-
	50	38444.0651	620.50	$2^-3^-4^+$	**40078.0983**	348.70	1^+4^+	39749.0757	424.25	1^+4^+	36447.2634	645.37	$1^-2^-3^-$
0.01	25	38026.4154	657.36	2^-3^-	**39525.2027**	425.96	1^+4^+	39240.9035	453.67	1^+4^+	37489.6489	742.56	2^-3^-
	50	36864.1232	555.40	$2^-3^-4^+$	**38332.2087**	312.35	1^+4^+	38000.8944	358.93	1^+4^+	34915.4010	565.36	$1^-2^-3^-$
0.001	25	35546.6995	551.64	2^-3^-	**36827.5418**	352.78	1^+4^+	36538.8560	340.16	1^+4^+	35095.0390	629.61	2^-3^-
	50	31947.2385	360.70	$2^-3^-4^+$	**32899.2694**	206.74	1^+4^+	32593.2603	203.93	1^+4^+	30253.8147	356.94	$1^-2^-3^-$

5 Experiments

In this work, we evaluate multi-objective evolutionary algorithms with our new fitness function, using extensive experimentation on several benchmarks under different chance constraint settings.

5.1 Experimental Settings

This work uses the six benchmarks in recent literature for the knapsack problem with stochastic profits [24]. These instances have different problem sizes $n \in \{100, 300, 500\}$ and types. This includes instances with bounded and strongly

Table 2. Experimental Results using Hoeffding bound

		GSEMO (1)			GSEMO+Filtering (2)			NSGA-II (3)			MOEA/D (4)		
α	δ	mean	std	stat	mean	std	stat	mean	std	stat	mean	std	stat
uncorr-100 (B = 2407)													
0.1	25	10987.3004	75.77	2^-	**11042.7989**	0.00	$1^+3^+4^+$	10988.0942	65.74	2^-	10947.9756	100.78	2^-
	50	10778.0078	60.20	4^+	**10821.5978**	0.00	3^+4^+	10749.3829	62.98	2^-4^+	10626.0009	135.69	$1^-2^-3^-$
0.01	25	10896.5878	74.59	2^-	**10951.1744**	0.00	$1^+3^+4^+$	10897.6552	64.40	2^-	10857.9983	99.27	2^-
	50	10596.7649	57.61	4^+	**10638.3488**	0.00	3^+4^+	10569.2343	60.55	2^-4^+	10449.1872	132.88	$1^-2^-3^-$
0.001	25	10826.9815	73.69	2^-	**10880.8684**	0.00	$1^+3^+4^+$	10828.2588	63.38	2^-	10788.9563	98.11	2^-
	50	10457.6924	55.62	4^+	**10497.7369**	0.00	3^+4^+	10431.0015	58.70	2^-4^+	10313.5134	130.75	$1^-2^-3^-$
strong-100 (B = 4187)													
0.1	25	8463.1467	127.62	2^-4^+	**8561.1573**	85.45	1^+4^+	8537.4284	83.34	4^+	8359.7780	89.80	$1^-2^-3^-$
	50	8278.6327	90.26	4^+	**8332.4692**	64.48	4^+	8289.8107	82.34	4^+	7931.1766	215.29	$1^-2^-3^-$
0.01	25	8369.3409	122.27	2^-4^+	**8464.4621**	81.51	1^+4^+	8440.2243	79.58	4^+	8268.8904	86.13	$1^-2^-3^-$
	50	8086.8101	82.32	2^-4^+	**8139.0049**	58.62	1^+4^+	8097.1159	74.76	4^+	7755.8441	205.27	$1^-2^-3^-$
0.001	25	8297.3868	118.20	2^-4^+	**8390.2653**	78.50	1^+4^+	8365.6369	76.71	4^+	8199.1499	83.37	$1^-2^-3^-$
	50	7939.6195	76.38	2^-4^+	**7990.5545**	54.16	1^+4^+	7949.3455	69.17	4^+	7621.3067	197.74	$1^-2^-3^-$
uncorr-300 (B = 6853)													
0.1	25	33863.1544	205.08	2^-	**34212.8177**	146.92	$1^+3^+4^+$	33925.6167	347.58	2^-	33732.3132	353.55	2^-
	50	33428.2570	259.29	2^-4^+	**33821.1723**	157.73	$1^+3^+4^+$	33568.9013	225.09	2^-4^+	31979.7768	639.54	$1^-2^-3^-$
0.01	25	33708.5096	203.93	2^-	**34055.7967**	145.63	$1^+3^+4^+$	33769.9207	345.68	2^-	33578.1622	351.09	2^-
	50	33119.8295	255.94	2^-4^+	**33507.4493**	154.52	$1^+3^+4^+$	33258.2607	222.13	2^-4^+	31683.9358	633.24	$1^-2^-3^-$
0.001	25	33589.8465	203.05	2^-	**33935.3102**	144.65	$1^+3^+4^+$	33650.4510	344.23	2^-	33459.8965	349.19	2^-
	50	32883.1649	253.39	2^-4^+	**33266.7212**	152.07	$1^+3^+4^+$	33019.9624	219.86	2^-4^+	31456.9291	628.42	$1^-2^-3^-$
strong-300 (B = 13821)													
0.1	25	23744.0892	430.47	2^-3^-	**24300.5479**	214.96	1^+4^+	24255.8224	290.79	1^+4^+	23269.1109	409.90	2^-3^-
	50	23462.9510	329.95	$2^-3^-4^+$	**23997.1125**	215.94	1^+4^+	23864.2031	225.53	1^+4^+	22197.4512	677.59	$1^-2^-3^-$
0.01	25	23603.7492	424.55	2^-3^-	**24152.7138**	210.67	1^+4^+	24105.8864	283.66	1^+4^+	23137.0365	404.26	2^-3^-
	50	23181.2241	317.94	$2^-3^-4^+$	**23700.6927**	206.31	1^+4^+	23565.9430	215.36	1^+4^+	21945.2578	660.78	$1^-2^-3^-$
0.001	25	23496.0973	420.02	$2^-3^-4^+$	**24039.3181**	207.29	1^+4^+	23990.8364	278.21	1^+4^+	23035.6923	399.94	$1^-2^-3^-$
	50	22965.0474	308.79	$2^-3^-4^+$	**23473.2418**	198.96	1^+4^+	23337.0800	207.65	1^+4^+	21751.7430	648.02	$1^-2^-3^-$
uncorr-500 (B = 11243)													
0.1	25	56985.6975	747.83	2^-3^-	**58337.8820**	310.84	$1^+3^+4^+$	57820.1034	646.79	$1^+2^-4^+$	56914.4076	780.54	2^-3^-
	50	56509.1689	856.18	$2^-3^-4^+$	**57934.0703**	312.74	1^+4^+	57362.7101	563.66	1^+4^+	52623.4475	1318.10	$1^-2^-3^-$
0.01	25	56790.6294	745.47	2^-3^-	**58137.6851**	309.25	$1^+3^+4^+$	57621.7379	644.06	$1^+2^-4^+$	56719.9375	777.75	2^-3^-
	50	56119.2292	849.61	$2^-3^-4^+$	**57533.3999**	309.31	1^+4^+	56966.3749	559.02	1^+4^+	52256.5983	1309.89	$1^-2^-3^-$
0.001	25	56640.9485	743.66	2^-3^-	**57984.0686**	308.03	$1^+3^+4^+$	57469.5268	641.97	$1^+2^-4^+$	56570.7153	775.61	2^-3^-
	50	55820.0179	844.58	$2^-3^-4^+$	**57226.0199**	306.67	1^+4^+	56662.2594	555.48	1^+4^+	51975.1049	1303.59	$1^-2^-3^-$
strong-500 (B = 22223)													
0.1	25	38739.7417	688.50	2^-3^-	**40301.3374**	447.34	1^+4^+	40018.5580	488.42	1^+4^+	38178.8224	775.66	2^-3^-
	50	38280.9153	613.70	$2^-3^-4^+$	**39897.8123**	344.90	1^+4^+	39568.7401	417.34	1^+4^+	36288.1082	636.21	$1^-2^-3^-$
0.01	25	38563.3178	680.78	2^-3^-	**40109.3511**	442.05	1^+4^+	39826.4698	479.77	1^+4^+	38008.5898	767.47	2^-3^-
	50	37930.8422	599.17	$2^-3^-4^+$	**39510.9695**	336.77	1^+4^+	39181.4670	402.63	1^+4^+	35947.7389	616.91	$1^-2^-3^-$
0.001	25	38427.9430	674.86	2^-3^-	**39962.0501**	437.99	1^+4^+	39679.0754	473.17	1^+4^+	37877.9658	761.19	2^-3^-
	50	37662.2216	588.08	$2^-3^-4^+$	**39214.1346**	330.57	1^+4^+	38884.3020	391.45	1^+4^+	35686.9510	602.80	$1^-2^-3^-$

correlated profit and weight values (denoted as strong-100, strong-300 and strong-500) and instances with uncorrelated profit and weight values (denoted as uncorr-100, uncorr-300 and uncorr-500). These problem instances are derived from benchmarks for the deterministic knapsack problem. When adapting them to have stochastic profits, we consider that profits are uniformly distributed as $p_i \in [\mu_i - \delta, \mu_i + \delta]$ where the profit values in the benchmarks give the expected profit μ_i. We consider each benchmark under two uncertainty levels, setting the dispersion of the profits as $\delta \in \{25, 50\}$.

We evaluate the performance of algorithms with different objective formulations by considering the results they generate for each benchmark setting for several α values, $\alpha \in \{0.1, 0.01, 0.001\}$. Each algorithm uses 10 million fitness evaluations in a single execution, which allows them to evaluate the same number of solutions despite the differences in the number of iterations.

Since the fitness function allows the multi-objective optimisation algorithms to run independently of specific α values, after one execution, we can get the maximal profit for different α values. We present the results considering the summary of 30 independent runs of each algorithm.

We test for the statistical significance validity of the results using the Kruskal-Wallis test with 95% confidence with the Bonferroni post-hoc statistical procedure. Let X be the number given for the methods in the table header; the statistical comparison X^+ or X^- indicates that the method in the column outperforms X or vice versa. If there is no significant difference between the two methods, respective numbers do not appear. The maximal profits from each method are given as mean, std and stat, representing the mean, standard deviation and statistical comparison with results from other methods, respectively. Furthermore, each row indicates the highest mean maximal profit in bold text, comparing the mean values given by the four methods.

5.2 Experimental Results

First, we consider bi-objective optimisation of the problem instances where profits have the same dispersion. Table 1 presents the $\hat{P}_{\text{Cheb}}$ results for $\delta = 25, 50$. When considering the results from the prevailing algorithms: GSEMO, NSGA-II and MOEA/D, we can see different performances based on the problem size. For problem instances with 100 items (i.e.: uncorr-100 and strong-100), GSEMO produces better results over MOEA/D for all settings and some cases of NSGA-II. However, when it comes to instances with 300 or more items, NSGA-II gives the best results compared to GSEMO and MOEA/D for overall settings. However, when considering the results when using the proposed Filtering method with GSEMO, we can see those results are significant across most of the settings. The statistical comparisons also confirm that GSEMO with the Filtering method inevitably produces better results than running the standard GSEMO alone. Moreover, we can see that GSEMO+Filtering results are superior to MOEA/D results in all the settings.

GSEMO+Filtering and NSGA-II can be identified as the two best algorithms according to $\hat{P}_{\text{Cheb}}$ results in Table 1. While the mean values of GSEMO+Filtering are higher than the mean values of NSGA-II, the statistical comparison between the two methods gives more insights. We can see that results for the strongly correlated and bounded instances (strong-100, -300 and -500) from both instances do not show significant differences. However, for uncorrelated instances (with 100, 300 and 500 elements), we can see that GSEMO+Filtering is capable of producing better results that show significantly higher than NSGA-II results according to statistical tests.

The experimental results concerning $\hat{P}_{\text{Hoef}}$ are given in Table 1.The difference between $\hat{P}_{\text{Cheb}}$ and $\hat{P}_{\text{Hoef}}$ results is, $\hat{P}_{\text{Hoef}}$ gives higher results for $\alpha = 0.001$ and vice versa for $\alpha = 0.1$ and 0.01. GSEMO produces better results than NSGA-II and MOEA/D in a lesser number of settings in $\hat{P}_{\mathrm{Hoef}}$ results than it does in $\hat{P}_{\mathrm{Cheb}}$ results. Statistical significance shows that GSEMO+Filtering results are not inferior to other methods across all settings. NSGA-II produce results that are similar to GSEMO+Filtering results for strong-100, -300 and -500 instances. However, for uncorrelated problem instances, GSEMO+Filtering gives better results over NSGA-II, especially when $\delta = 25$. Moreover, similar to results in Table 1, the highest mean value and lowest standard deviation in $\hat{P}_{\mathrm{Hoef}}$ results are demonstrated by GSEMO+Filtering.

When Filtering is applied in an interim population, the solutions with valid confidence intervals remain, therefore the population do not lose its quality. Then, the optimisation process improves upon these solutions without the negative impacts of having a larger population with a similar quality. Among the results, both GSEMO+Filtering and NSGA-II show great results. However, considering mean, standard deviation and statistical comparisons, GSEMO+Filtering is the most promising approach to solve the types of problem instances focused in this study.

6 Conclusions

We examined popular multi-objective evolutionary approaches to solve the profit chance-constrained knapsack problem. Thereby, we introduced a bi-objective fitness evaluation for evolutionary algorithms to address this problem. This fitness function evaluates the solutions irrespective of the required confidence level in the profit. Therefore, the outcome of evolutionary methods gives us a population that includes solutions providing the best profit value for different confidence levels. Thus, our approach only requires us to decide on the required confidence levels after running algorithms. It also eliminates the requirement of running algorithms multiple times to obtain results for different confidence levels.

As a key algorithmic component, we introduced a filtering method to GSEMO to filter the population at regular intervals of fitness evaluations. It keeps only the solutions with a valid α interval in the interim populations, enabling the new offspring solutions in the next generations to improve upon these solutions. The experimental investigations show that the results from the filtering method integrated with GSEMO are often better than those obtained by standard GSEMO, NSGA-II and MOEA/D. The final population produced by our algorithms allow us to make informed decisions and examine solution quality concerning the expected profit versus the associated risks with their implementation.

Acknowledgements. This work has been supported by the Australian Research Council (ARC) through grant FT200100536, and with supercomputing resources provided by the Phoenix HPC service at the University of Adelaide.

References

1. Abe, Y., Ogura, M., Tsuji, H., Miura, A., Adachi, S.: Resource and network management for satellite communications systems: A chance-constrained approach. IFAC **53**, 3304–3309 (2020)
2. Ahouei, S.S., de Nobel, J., Neumann, A., Bäck, T., Neumann, F.: Evolving reliable differentiating constraints for the chance-constrained maximum coverage problem. In: Genetic and Evolutionary Computation Conference, GECCO 2024. ACM (2024), to appear
3. Assimi, H., Harper, O., Xie, Y., Neumann, A., Neumann, F.: Evolutionary biobjective optimization for the dynamic chance-constrained knapsack problem based on tail bound objectives. In: 24th European Conference on Artificial Intelligence. vol. 325, pp. 307–314. IOS Press (2020)
4. Baumeister, D., Boes, L., Laußmann, C.: Time-constrained participatory budgeting under uncertain project costs. In: Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI). pp. 74–80 (2022)
5. Chauhan, A., Baranwal, M., Basumatary, A.: PowRL: A reinforcement learning framework for robust management of power networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 14757–14764 (2023)
6. Coello, C., Van Veldhuizen, D., Lamont, G.: Evolutionary Algorithms for Solving Multi-Objective Problems. Genetic Algorithms and Evolutionary Computation, Springer US (2013)
7. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. John Wiley & Sons (2001)
8. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002)
9. Deb, K., Gupta, S., Daum, D., Branke, J., Mall, A.K., Padmanabhan, D.: Reliability-based optimization using evolutionary algorithms. IEEE Transactions on Evolutionary Computation **13**(5), 1054–1074 (2009)
10. Doerr, B., Doerr, C., Neumann, A., Neumann, F., Sutton, A.M.: Optimization of chance-constrained submodular functions. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020. pp. 1460–1467. AAAI Press (2020)
11. Don, T.P., Neumann, A., Neumann, F.: The chance constrained travelling thief problem: Problem formulations and algorithms. In: Genetic and Evolutionary Computation Conference, GECCO 2024. ACM (2024), to appear
12. Doskoc, V., Friedrich, T., Göbel, A., Neumann, A., Neumann, F., Quinzan, F.: Non-monotone submodular maximization with multiple knapsacks in static and dynamic settings. In: 24th European Conference on Artificial Intelligence. vol. 325, pp. 435–442 (2020)
13. Geng, X., Xie, L.: Data-driven decision making in power systems with probabilistic guarantees: Theory and applications of chance-constrained optimization. Annual Reviews in Control **47**, 341–363 (2019)
14. Giel, O.: Expected runtimes of a simple multi-objective evolutionary algorithm. In: The 2003 Congress on Evolutionary Computation. vol. 3, pp. 1918–1925 Vol.3 (2003)
15. He, F., Shao, G.: An evolutionary algorithm for uncertain optimization problems. In: 2009 International Conference on Information Engineering and Computer Science. pp. 1–4. IEEE (2009)
16. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack problems. Springer (2004)

17. Kepaptsoglou, K., Fountas, G., Karlaftis, M.G.: Weather impact on containership routing in closed seas. Transportation Research Part C: Emerging Technologies **55**, 139–155 (2015)
18. Liu, B., Zhang, Q., Fernández, F.V., Gielen, G.G.E.: An efficient evolutionary algorithm for chance-constrained bi-objective stochastic optimization. IEEE Transactions on Evolutionary Computation **17**(6), 786–796 (2013)
19. Loughlin, D.H., Ranjithan, S.R.: Chance-constrained genetic algorithms. In: Genetic and Evolutionary Computation Conference, GECCO 1999. p. 369–376. Morgan Kaufmann Publishers Inc. (1999)
20. Masutomi, K., Nagata, Y., Ono, I.: An evolutionary algorithm for black-box chance-constrained function optimization. Journal of Advanced Computational Intelligence and Intelligent Informatics **17**(2), 272–282 (2013)
21. Mitzenmacher, M., Upfal, E.: Probability and Computing: Randomized Algorithms and Probabilistic Analysis. Cambridge University Press (2005)
22. Neumann, A., Bossek, J., Neumann, F.: Diversifying greedy sampling and evolutionary diversity optimisation for constrained monotone submodular functions. In: GECCO '21: Genetic and Evolutionary Computation Conference 2021. pp. 261–269. ACM (2021)
23. Neumann, A., Neumann, F.: Optimising monotone chance-constrained submodular functions using evolutionary multi-objective algorithms. In: Parallel Problem Solving from Nature - PPSN XVI - 16th International Conference, PPSN 2020, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12269, pp. 404–417. Springer (2020)
24. Neumann, A., Xie, Y., Neumann, F.: Evolutionary algorithms for limiting the effect of uncertainty for the knapsack problem with stochastic profits. In: PPSN XVII. pp. 294–307. Springer, Cham (2022)
25. Neumann, F., Pourhassan, M., Roostapour, V.: Analysis of Evolutionary Algorithms in Dynamic and Stochastic Environments, chap. 7, pp. 323–357. Springer (2020)
26. Neumann, F., Sutton, A.M.: Runtime analysis of the $(1+1)$ evolutionary algorithm for the chance-constrained knapsack problem. In: FOGA '19. p. 147–153. FOGA '19, ACM (2019)
27. Neumann, F., Witt, C.: Runtime analysis of single- and multi-objective evolutionary algorithms for chance constrained optimization problems with normally distributed random variables. In: Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI). pp. 4800–4806 (7 2022)
28. Pathiranage, I.H., Neumann, F., Antipov, D., Neumann, A.: Effective 2- and 3-objective MOEA/D approaches for the chance constrained knapsack problem. In: Genetic and Evolutionary Computation Conference, GECCO 2024. ACM (2024), to appear
29. Pathiranage, I.H., Neumann, F., Antipov, D., Neumann, A.: Using 3-objective evolutionary algorithms for the dynamic chance constrained knapsack problem. In: Genetic and Evolutionary Computation Conference, GECCO 2024. ACM (2024), to appear
30. Perera, K., Neumann, A.: Multi-objective evolutionary algorithms with sliding window selection for the dynamic chance-constrained knapsack problem. In: Genetic and Evolutionary Computation Conference, GECCO 2024. ACM (2024), to appear
31. Reid, W., Neumann, A., Ratcliffe, S., Neumann, F.: Advanced mine optimisation under uncertainty using evolution. In: GECCO 2021: Genetic and Evolutionary Computation Conference, Companion Volume. pp. 1605–1613. ACM (2021)

32. Roostapour, V., Neumann, A., Neumann, F.: On the performance of baseline evolutionary algorithms on the dynamic knapsack problem. In: Parallel Problem Solving from Nature – PPSN XV. pp. 158–169. Springer International Publishing, Cham (2018)
33. Roostapour, V., Neumann, A., Neumann, F.: Single- and multi-objective evolutionary algorithms for the knapsack problem with dynamically changing constraints. Theor. Comput. Sci. **924**, 129–147 (2022)
34. Roostapour, V., Neumann, A., Neumann, F., Friedrich, T.: Pareto optimization for subset selection with dynamic cost constraints. Artif. Intell. **302**, 103597 (2022)
35. Shi, F., Yan, X., Neumann, F.: Runtime analysis of simple evolutionary algorithms for the chance-constrained makespan scheduling problem. In: PPSN XVII. pp. 526–541. Springer (2022)
36. Stimson, M., Reid, W., Neumann, A., Ratcliffe, S., Neumann, F.: Improving confidence in evolutionary mine scheduling via uncertainty discounting. In: IEEE Congress on Evolutionary Computation, CEC 2023. pp. 1–10. IEEE (2023)
37. Wang, A.J., Williams, B.C.: Chance-constrained scheduling via conflict-directed risk allocation. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. p. 3620–3627. AAAI'15, AAAI Press (2015)
38. Xie, Y., Harper, O., Assimi, H., Neumann, A., Neumann, F.: Evolutionary algorithms for the chance-constrained knapsack problem. In: Genetic and Evolutionary Computation Conference, GECCO 2019. p. 338–346. ACM (2019)
39. Xie, Y., Neumann, A., Neumann, F.: Specific single- and multi-objective evolutionary algorithms for the chance-constrained knapsack problem. In: Genetic and Evolutionary Computation Conference, GECCO 2020. p. 271–279. ACM (2020)
40. Xie, Y., Neumann, A., Neumann, F.: Heuristic strategies for solving complex interacting stockpile blending problem with chance constraints. In: Genetic and Evolutionary Computation Conference, GECCO 2021. p. 1079–1087. ACM (2021)
41. Xie, Y., Neumann, A., Neumann, F., Sutton, A.M.: Runtime analysis of RLS and the (1+1) EA for the chance-constrained knapsack problem with correlated uniform weights. In: Genetic and Evolutionary Computation Conference, GECCO 2021. p. 1187–1194. ACM (2021)
42. Yan, X., Neumann, A., Neumann, F.: Sampling-based pareto optimization for chance-constrained monotone submodular problems. In: Genetic and Evolutionary Computation Conference, GECCO 2024. ACM (2024), to appear
43. Yan, X., Neumann, A., Neumann, F.: Sliding window bi-objective evolutionary algorithms for optimizing chance-constrained monotone submodular functions. In: Parallel Problem Solving from Nature XVIII, PPSN 2024 (2024), to appear
44. Zhang, Q., Li, H.: MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation **11**(6), 712–731 (2007)