A Precedence Constrained Knapsack Problem with Uncertain Item Weights for Personalized Learning Systems

Ayse Aslan, Evrim Ursavas, Ward Romeijnders

PII: S0305-0483(22)00186-4
DOI: https://doi.org/10.1016/j.omega.2022.102779
Reference: OME 102779

To appear in: *Omega*

Received date: 8 October 2021
Accepted date: 29 September 2022

Please cite this article as: Ayse Aslan, Evrim Ursavas, Ward Romeijnders, A Precedence Constrained Knapsack Problem with Uncertain Item Weights for Personalized Learning Systems, *Omega* (2022), doi: https://doi.org/10.1016/j.omega.2022.102779

Highlights

- Studies a novel educational context problem of personalized learning environments.
- Presents a new precedence-constrained knapsack model with uncertain item weights.
- Provides efficient approximate cutting plane methods with Taylor approximations.
- Provides effective pre-processing procedures and cover cuts.

A Precedence Constrained Knapsack Problem with Uncertain Item Weights for Personalized Learning Systems

Ayse Aslan[1], Evrim Ursavas[2], and Ward Romeijnders[2]

[1]*School of Engineering and The Built Environment, Edinburgh Napier University, Edinburgh EH10 5DT, United Kingdom*

[2]*Department of Operations, University of Groningen, Groningen 9747 AD, The Netherlands*

Email addresses: a.aslan@napier.ac.uk (Ayse Aslan, corresponding author), e.ursavas@rug.nl (Evrim Ursavas), w.romeijnders@rug.nl (Ward Romeijnders),

September 30, 2022

Abstract

This paper studies a unique precedence constrained knapsack problem in which there are two methods available to place an item in the knapsack. Whether or not an item weight is uncertain depends on which one of the two methods is selected. This knapsack problem models students' decisions on choosing subjects to study in hybrid personalized learning systems in which students can study either under teacher supervision or in an unsupervised self-study mode by using online tools. We incorporate the uncertainty in the problem using a chance-constrained programming framework. Under the assumption that uncertain item weights are independently and normally distributed, we focus on the deterministic reformulation in which the capacity constraint involves a nonlinear and convex function of the decision variables. By using the first-order linear approximations of this function, we propose an exact cutting plane method that iteratively adds feasibility cuts. To supplement this, we develop novel approximate cutting plane methods that converge quickly to high-quality feasible solutions. To improve the computational efficiency of our methods, we introduce new pre-processing procedures to eliminate items beforehand and cover cuts to refine the feasibility space. Our computational experiments on small and large problem instances show that the optimality gaps of our approximate methods are very small overall, and that they are even able to find solutions with no optimality gaps as the number of items increases in the instances. Moreover, our experiments demonstrate that our pre-processing methods are particularly effective when the precedence relations are dense, and that our cover cuts may significantly speed up our exact cutting plane approach in challenging instances.

Keywords— Knapsack Problems; Chance-constrained Programming; Cutting Plane Methods; Personalized Learning; OR in Education; Precedence Constraints

1 Introduction

Education is a major service sector in many countries, with OECD countries spending on average 5% of their gross domestic products on it (NCES, 2020). For years, educational services have been provided to students under the one-size-fits-all approach (Ohanian, 1999) that offers standardized learning paths for everyone in a certain group (e.g., age, level). However, these services are undergoing a world-wide transition from the one-size-fits-all approach towards *personalized learning* (PL) models (UNESCO, 2017; European Commission, 2019). In fact, various PL models, in which students have a certain flexibility to customize their learning processes, are already being implemented, see Eiken (2011), Kannan et al. (2012) and Prain et al. (2013), for examples from Europe, US, and Australia, respectively.

In contrast to the one-size-fits-all approach, in PL students take an active role in setting their learning goals and determining how to achieve them. In other words, they compose their personal learning plans; they select what they want to learn, when they want to learn and how they want to learn (Pane et al., 2017). In order to facilitate this active learning, apart from instructing various learning activities, teachers guide the learning progress of their students, for example, by holding coaching meetings with their students (Kunskapsskolen, 2021). Technology also plays an important role in facilitating the flexibility in the PL services, for example, by allowing the educational services to be provided digitally (e.g., `https://www.khanacademy.org/`). In combination with flexibly used physical resources (i.e., teachers, classrooms and time blocks) these create a flexible learning environment that facilitates personalization for every student (Pane et al., 2017).

Given the flexibility that PL offers to students in customizing their learning, the study choices available for students are increasing in PL environments. In customizing their individual learning plans, students can make extensive choices about the content, method and structure of their learning processes. Moreover, the content of subjects is being divided into increasingly smaller, interconnected units, making it possible for students to personalize their plans down to the atomic knowledge units of the subjects (Lightfoot, 2006). Due to this higher flexibility for students to select how and what they want to learn, their learning decisions can become highly complex. For this reason, students in PL can benefit from systematic decision-making tools in customizing their learning process.

This decision problem of a student in selecting which subjects, or subject units, to study during a specific academic period can be cast as a precedence constrained knapsack problem (PCKP) where the set of items to be placed in the knapsack represents the subjects and the precedence constraints represent the prerequisite relations among them, modeling that some subjects cannot be studied before knowledge about others is obtained. The item weights represent the number of hours that the student needs to study the subjects, while the knapsack capacity represents the total number of hours that the student has available for studying in the period. Finally, the item profits represent the student's personal reward for studying a subject, for example, the number of credits earned, or any other measure reflecting the student's personal preferences and goals.

In this paper, we provide a PCKP model with uncertainty in item weights to represent the students' problem of selecting subjects in PL environments. Uncertainty is an important aspect of our model, since in PL technology plays an important role in complementing teacher-supervised instruction with online learning tools. In particular, the time that students need for studying using online learning tools in an unsupervised self-study mode is uncertain. In practice, this uncertainty can be due to student-specific factors such as self-discipline, self-motivation and learning pace, which may vary a lot among students. On the other hand, we acknowledge

that the number of hours needed for studying under teacher supervision is mostly known (e.g., fixed number of in-class meetings each week). Hence, we consider that the item weights corresponding to teacher-supervised instruction activities are deterministic, whereas the weights corresponding to self-study activities by online tools are modeled using random parameters in our knapsack problem. Students may select whether to study unsupervised or under the supervision of a teacher for any subject they choose to study. We assume that this learning method selection does not affect the rewards that students will obtain by studying a subject. For example, if the rewards correspond to the credits earned by completing the subjects, the method employed in completing subjects do not play a role in the rewards. As pointed out in Kim et al. (2014), online learning can be very challenging for many students. Therefore, we consider that students prefer teacher-supervised studying over self-study because they expect that the time they need for studying a subject unsupervised will be at least as lengthy as studying the subject under the supervision of a teacher. The issue, however, is that teacher-supervised instruction may not always be available, due to the limited capacity of teachers. Therefore, students are forced to carry out at least a proportion $R \in [0, 1]$ of their learning activities unsupervised.

We note that the level for R is to be determined by external parties, for example, by large public schools who have to serve many students in teacher-led instruction activities with a limited teaching staff level. Schools with limited teaching capacity should carefully determine R, to make sure that every student has access to teacher-supervised learning at the time they need it. An additional challenge here is that demands for learning activities change over time, since students in PL learn at their own pace, and as a result, teacher activities are organized flexibly in real-time to satisfy learning demands, see Aslan et al. (2020). Finally, we point to a concrete example from education where an enforcement of unsupervised learning activities can be inevitable. Currently, the COVID-19 pandemic has turned the schools into hybrid learning environments in which students can have regular in-class meetings with their teachers only partially and have to rely on remote learning up to some extent, due to the restrictions on the capacity for in-class teaching. These schools then need to set a limit to the activities that can be carried on-site.

Our paper uses a chance-constrained programming framework (Charnes & Cooper, 1959). Similar to Lu et al. (2021) and Qiu et al. (2022), we use the chance constraint to ensure resource sufficiency in the probabilistic sense. That is, we restrict the probability that the total study time exceeds a student's available study time. Moreover, we assume that the uncertain item weights are independently [1] and normally [2] distributed. The normality assumption on the distribution of random item weights is commonly adopted in the stochastic knapsack literature (Weintraub & Vera, 1991; Han et al., 2016; Joung & Lee, 2020; Lyu et al., 2022). Additionally, normal distributions are commonly used in education for modeling processes and data, for example, for modeling achievement levels of a student population on a course (Bloom, 1968), and learning pace profiles of students (Aslan et al., 2020). By exploiting the normality assumption, we reformulate the probabilistic knapsack capacity constraint as a deterministic one, in order to make use of integer programming (IP) solvers. However, due to the variances involving the distributions of uncertain item weights, the resulting deterministic capacity constraint involves a convex but nonlinear function of the decision variables, and due to the nonlinearity of the capacity constraint, as also stated in Han et al. (2016), even under the normality assumption, the use of IP solvers can be only practical for stochastic knapsack problem instances with a very limited number of items. For this reason, various approaches are proposed in the literature for overcoming the nonlinearity. Joung & Lee (2020) proposed a robust optimization–based heuristic, by using ellipsoid uncertainty sets to translate the nonlinear capacity constraint into an exponential number of linear capacity constraints. Obviously, piece-wise

linear approximations may also be utilized for this, as in Lin (2009). Weintraub & Vera (1991) proposed a cutting plane algorithm for chance-constrained linear maximization problems under normally distributed coefficients in the constraints. This algorithm iteratively solves a linear model and uses the solution produced at an iteration to formulate cuts that will refine the feasibility space in the next iteration. In the spirit of Weintraub & Vera (1991), this paper proposes cutting plane methods that refine the feasibility space gradually for solving our precedence constrained knapsack problem with normally distributed uncertain item weights.

The cutting plane method of Weintraub & Vera (1991) is a supporting hyperplane algorithm that requires finding an interior point and a boundary point of the feasibility space at each iteration to produce feasibility cuts. Our cutting plane methods resolve these issues by using the first-order linear Taylor approximations of the nonlinear and convex function involved in the capacity constraint to produce feasibility cuts at each iteration to refine the feasibility space gradually. One of our cutting plane methods is exact; at the iteration that it finds a feasible solution, the solution obtained will be optimal. It is worth mentioning that this exact method can be considered as an outer approximation method (Duran & Grossmann, 1986), which is proposed for solving convex mixed-integer nonlinear programs, since our method is also based on exploiting the convexity of the nonlinear functions in the mixed-integer nonlinear programs to formulate equivalent linear programs with tangent planes.

Evidently, this exact method can be computationally burdensome, as it may require many iterations until it finds a feasible solution. To supplement this method, we propose novel approximate cutting plane methods that provide high-quality feasible solutions in a few iterations. Different from the exact method, our approximate methods consider approximate versions of the nonlinear and convex function in the capacity constraint and use their first-order linear approximations to derive approximate feasibility cuts. The scheme we use for approximating this nonlinear and convex function in our approximate methods exploits the fact that there is an equivalent concave expression of this function and considers the convex combinations of the original convex function and its equivalent concave version. The higher the weight of the concave version in the combination, the faster the convergence can be in obtaining feasible solutions. However, this might reduce the solution quality. On the other hand, the higher the weight of the original convex function in the combination, the closer the approximate method resembles the exact cutting plane method. One of our approximate methods that only considers the concave version requires at most two iterations to solve any problem instance. As the feasibility cut that this method uses always approximates the feasibility region from the inside, it has links to the inner approximation methods proposed for mixed-integer nonlinear programs (Hijazi et al., 2013).

Using extensive computational experiments on both small and large problem instances, we show that our approximate cutting plane methods are able to quickly find feasible solutions with very small optimality gaps. Moreover, as the number of items in instances increases these optimality gaps approach zero. We find that only when the variances of the uncertain item weights are very high and/or the means of the uncertain item weights are negatively correlated to their variances, then the optimality gaps of our approximate methods may be relatively higher. In our experiments, we also demonstrate the efficiency of our approximate cutting plane methods in solving the chance-constrained binary knapsack problem with independent and normally distributed weights (Han et al., 2016; Joung & Lee, 2020), which is a special case of our problem.

Additionally, we propose pre-processing procedures that eliminate items beforehand and cover cuts to improve the computational efficiency of the cutting plane methods, which both exploit the characteristics of our unique precedence constrained knapsack problem with uncertain item weights. Different from the item elimination procedures and cover cuts proposed earlier for PCKP, we do not solely exploit the precedence constraints

but also extensively use lower and upper bounds on the cardinalities of optimal solutions, which we find with the help of heuristic solutions, and exploit distributional information on uncertain item weights. In our computational experiments, we demonstrate the significant benefit of adding cover cuts in increasing the convergence speed of the exact cutting plane method, especially in the difficult instances that require very long running times, and we show that pre-processing becomes highly effective when the precedence relations among items are high in density, and this effectiveness becomes more visible in instances with a large number of items.

Section 2 presents the literature related to our precedence constrained knapsack problem. In Section 3, we formulate the problem and provide a chance-constrained programming formulation for it. Section 4 presents the pre-processing procedures, the exact and approximate cutting plane methods and the cover cuts proposed for our problem. In Section 5, we conduct computational experiments to demonstrate the efficiency of the proposed methods and to compare their performance with several benchmark methods. Lastly, Section 6 concludes the paper.

2 Related Literature

The decision-making problems studied in education are heavily focused on the problems of one-size-fits-all learning systems (see Johnes (2015)). Most of these problems are deterministic timetabling problems (Pillay, 2014) pertaining to the one-size-fits-all approach which assigns learners to age-based groups and plans instruction for these alone. The literature also covers transportation and mechanism design problems in education, such as school bus routing and school choice problems (Smilowitz & Keppler, 2020). To the best of our knowledge, there are only a few studies that consider the decision problems of PL systems (Aslan et al., 2020; Kannan et al., 2012; Kristiansen et al., 2011). These three studies consider decision-making from the perspective of service providers (e.g., schools). Our paper is novel as it studies a decision-making problem of students. The problem we focus on is a stochastic precedence-constrained knapsack problem (PCKP) with uncertainty in item weights. In what follows, we review the related literature on PCKP and other knapsack problems studied under uncertainty.

PCKP is a generalization of the well-known 0-1 knapsack problem (KP) in which the given set of items I are subject to some precedence constraints (E. A. Boyd, 1993; Van de Leensel et al., 1999; Samphaiboon & Yamada, 2000; You & Yamada, 2007; Boland et al., 2012). Letting $\mathcal{E} \subseteq I \times I$ denote the set of all precedence relations, the edge $(i, j) \in \mathcal{E}$ imposes the precedence constraint that item j cannot be placed in the knapsack without item i. Given that each item $i \in I$ has a profit value p_i and a weight of w_i, PCKP aims for finding a subset of the items with maximum profit without exceeding the knapsack capacity B and violating any of the precedence relations in $\mathcal{E}$.

PCKP, which is shown to be an NP-complete problem by Garey & Johnson (1979), is not solely relevant in educational contexts (Haase et al., 1999); PCKP appears in a wide range of applications. For example, PCKP models the investment planning in strip mining (Johnson & Niemi, 1983), and it serves as a subproblem in the local access network design problem (Shaw & Cho, 1998) and in several resource-constrained multi-period scheduling problems in which tasks to be scheduled are subject to precedence constraints, such as the multi-period open-pit mine production scheduling problem (Samavati et al., 2018; Chicoisne et al., 2012), project scheduling (Araujo et al., 2020; Nansheng & Qichen, 2022), job scheduling (Kasapidis et al., 2021), maintenance scheduling (Yoo & Garcia-Diaz, 2008) and therapy scheduling (Nossack, 2022).

Deterministic PCKP formulations in which all problem parameters are assumed known have severe shortcomings for modeling many real-life applications in which there are uncertainties around the problem parameters. When uncertainty is involved, decision makers are often interested in avoiding risks, and therefore, stochastic PCKP formulations explicitly take this risk into account when determining the most profitable subset of items. For example, for modeling the open-pit mine production scheduling problem, due to the geological uncertainties present in the ore grades of blocks (Lagos et al., 2020), stochastic PCKP formulations which incorporate the uncertainty in item profits are better suited. Moreover, for modeling the subject selection decisions of students in personalized learning environments, which is the motivation of our paper, the uncertainties around the study times of the subjects should be incorporated in a stochastic PCKP formulation.

In the literature, stochastic formulations for different variants and generalizations of the 0-1 knapsack problem are emerging and in formulating these, stochastic programming, chance-constrained programming and robust optimization constitute the major modeling frameworks. Morton & Wood (1998) studied the 0-1 stochastic knapsack problem with a chance-constrained objective function for the case that item profits are random with known distributions. Kosuch & Lisser (2010), on the other hand, considered the case that item weights are uncertain and provided formulations within the frameworks of simple recourse stochastic and chance-constrained programming, for the 0-1 stochastic knapsack problem. Tönissen et al. (2017) used the two-stage stochastic programming framework to present a stochastic formulation for the multiple knapsack problem in which the knapsack capacities are subject to uncertainty. Monaci & Pferschy (2013) studied a robust knapsack problem with uncertain item weights in which weights belong to known intervals and the number of items whose weight differs from the nominal weight value is bounded by a constant. Recently, Caserta & Vos (2019) provided robust optimization formulations for the multiple-choice multidimensional knapsack problem. However, to the best of our knowledge, existing optimization approaches to PCKP take a deterministic standing and assume that there are no uncertainties in the problem parameters. Our paper provides a stochastic formulation for PCKP with uncertain item weights in a chance-constrained programming framework.

We note that cover cuts for PCKP are well studied in the literature (E. A. Boyd, 1993; Van de Leensel et al., 1999). These cuts exploit the precedence relations among items. In this paper, we extend the use of cover cuts to our precedence constrained knapsack problem with uncertain item weights. Different from the cuts proposed for PCKP, our cover cuts incorporate the distributional information on the uncertain item weights and exploit the lower and upper bounds we find on the cardinality of optimal solutions. In our computational experiments, we demonstrate the effectiveness of these in improving the computational efficiency of the exact cutting plane method.

3 Problem Description and Modeling

3.1 Problem Description

Consider a decision maker with a knapsack of capacity B who is presented a set of items I to place in the knapsack where each item $i \in I$ is associated with profit p_i. There are precedence relations among the items. Let $E \subseteq I \times I$ denote the set of immediate precedence relations among the items such that $(i, j) \in E$ represents the relation that item i is an *immediate predecessor* of item j and item j is an *immediate successor* of item i. The precedence relation $(i, j) \in E$ imposes the precedence constraint that item j cannot be placed in the

knapsack if item i is not placed. We note that due to the transitivity of precedence relations, the set of all precedence relations $\mathcal{E}$ is the transitive closure of E, and thus $(i,j) \in \mathcal{E}$ means item i is a *predecessor* of item j and item j is a *successor* of item i. We let $P(i) = \{j|(j,i) \in \mathcal{E}\}$ and $S(i) = \{j|(i,j) \in \mathcal{E}\}$ denote the *set of predecessors and successors*, respectively, of item i. We assume that the precedence relations are well-defined such that the precedence graph $G = (I, E)$ is a directed and acyclic graph (DAG).

There are two methods available to the decision maker for placing an item in the knapsack, which we refer to as the preferred (e.g., teacher-supervised instruction) and non-preferred (e.g., unsupervised self-study with online tools) methods. The method used affects the weights of the items placed in the knapsack; under the non-preferred method item weights are uncertain. Per item i, w_i denotes the deterministic weight of the item when it is placed with the preferred method and ξ_i is the random variable denoting the uncertain weight of the item when the non-preferred method is used. For each item i, the relation $\mathbf{E}[\xi_i] \geq w_i$ holds. An external party (e.g., a school) imposes the constraint upon the decision maker to use the non-preferred method in at least $R \in [0,1]$ proportion of the items to be placed in the knapsack. Then, the objective of the decision maker is to place the most profitable subset of items in the knapsack, while ensuring that the total weight of the items selected will not exceed the capacity of the knapsack with probability of at least $\alpha \in [0,1]$. Of course, the subset of items also needs to satisfy the precedence constraints imposed by E, and use the non-preferred method in at least a proportion R of the items.

3.2 Model

We use the binary decision variables $x_i, y_i \in \{0,1\}$, $\forall i \in I$, where the variable x_i determines whether item i is selected or not, whereas the variable y_i determines if the non-preferred method is used or not. When $x_i = 1$ and $y_i = 0$, then item i is placed in the knapsack with the preferred method, with the deterministic item weight w_i, and when $x_i = 1$ and $y_i = 1$, then the item will be placed with the non-preferred method with the uncertain item weight ξ_i. If $x_i = 0$, then item i is not selected in the knapsack, and we force that $y_i = 0$.

3.2.1 Chance-constrained Formulation

The following presents the chance-constrained integer programming model for our problem.

$$\text{Maximize} \quad \sum_{i \in I} p_i x_i \tag{1}$$

$$\text{subject to} \quad P\Big\{\sum_{i \in I} w_i x_i + \sum_{i \in I} (\xi_i - w_i) y_i \leq B\Big\} \geq \alpha \tag{2}$$

$$\sum_{i \in I} y_i \geq R \sum_{i \in I} x_i \tag{3}$$

$$x_i \geq x_j \qquad \forall (i,j) \in E \tag{4}$$

$$y_i \leq x_i \qquad \forall i \in I \tag{5}$$

$$x_i \in \{0,1\}, y_i \in \{0,1\} \qquad \forall i \in I. \tag{6}$$

Constraint (2) is the chance constraint imposing that the total weights of the items placed in the knapsack should not exceed the capacity B with at least probability $\alpha \in [0,1]$. Constraints (4) enforce the precedence

relations between items. Constraint (3) ensures that in at least a proportion R of the selected items, the non-preferred method is used, and constraint (5) links the decision variables x_i and y_i per item i such that when item i is not selected ($x_i = 0$), y_i cannot be equal to one. Finally, the objective in (1) represents the profit of the selected subset of items.

3.2.2 Deterministic Reformulation for Normally Distributed Uncertain Weights

We assume that for each item $i \in I$, ξ_i is normally distributed with a known mean ν_i and variance σ_i^2, independently from the other items. Therefore, also $\xi_i - w_i$ is a normally distributed random variable for any item $i \in I$ with mean $\mu_i := \nu_i - w_i$ and variance σ_i^2. Since we assume that $\mathbf{E}[\xi_i] \geq w_i$ for any item $i \in I$, it follows that μ_i and σ_i are non-negative values for every item i. In the rest of the paper, for each item $i \in I$, we refer to μ_i and σ_i^2 as the mean and variance of the uncertain weight of the item.

To reformulate the chance constraint in (2), we observe that $\sum_{i\in I} y_i(\xi_i - w_i) \sim \mathcal{N}(\mu^T y, y^T \Sigma y)$, where μ is the $|I|$-dimensional mean vector and Σ is the $|I| \times |I|$-dimensional covariance matrix whose element in the ith row and jth column gives the covariance of random variables ξ_i and ξ_j. Under the normality assumption, constraint (2) can be formulated as

$$P\Big\{\frac{\sum_{i\in I}(\xi_i - w_i)y_i - \mu^T y}{\sqrt{y^T\Sigma y}} \leq \frac{B - \sum_{i\in I} w_i x_i - \mu^T y}{\sqrt{y^T\Sigma y}}\Big\} \geq \alpha, \tag{7}$$

which then can be written as

$$\frac{B - \sum_{i\in I} w_i x_i - \mu^T y}{\sqrt{y^T\Sigma y}} \geq \Phi^{-1}(\alpha), \tag{8}$$

where Φ^{-1} is the inverse cumulative function of the standard normal distribution (see Prekopa (2003)). Since we assume that there is no correlation between ξ_i and ξ_j for any $i \in I$ and $j \in I, j \neq i$, the covariance matrix Σ will be a diagonal matrix of the form $\begin{bmatrix} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_{|I|}^2 \end{bmatrix}$. Thus, we have that $\sqrt{y^T\Sigma y} = \sqrt{\sum_{i\in I} \sigma_i^2 y_i^2}$. In the rest of the paper, we refer to the following deterministic formulation as the two-choice precedence constrained knapsack problem.

$$\text{2cPCKP}: \text{Maximize} \quad \sum_{i\in I} p_i x_i \tag{9}$$

$$\text{subject to} \quad \sum_{i\in I} w_i x_i + \sum_{i\in I} \mu_i y_i + \Phi^{-1}(\alpha)\sqrt{\sum_{i\in I} \sigma_i^2 y_i^2} \leq B \tag{10}$$

$$\sum_{i\in I} y_i \geq R \sum_{i\in I} x_i \tag{11}$$

$$x_i \geq x_j \qquad \forall (i,j) \in E \tag{12}$$

$$y_i \leq x_i \qquad \forall i \in I \tag{13}$$

$$x_i \in \{0,1\}, y_i \in \{0,1\} \qquad \forall i \in I. \tag{14}$$

It must be noted that when $\alpha = 0.5$, we have $\Phi^{-1}(0.5) = 0$, and thus this formulation reduces to a nominal one in which the uncertain weights ξ_i are replaced by their means. In the remainder of this paper we assume that $\alpha \geq 0.5$. Furthermore, we note that for $R = 0$ this formulation corresponds to the deterministic PCKP while for $R = 1$ it models the chance-constrained PCKP in which the weight of every item is subject to uncertainty. For this reason, in the rest of the paper, we refer to R as the uncertainty rate parameter.

It is worth emphasizing that this deterministic reformulation of the chance-constrained knapsack problem, which is a second-order cone program, can be interpreted as the robust counterpart of a robust knapsack problem with ellipsoidal uncertainty sets, as noted also in Calafiore & Ghaoui (2006) for linear programs.

4 Solution Methodology

4.1 Pre-Processing Procedures

To eliminate items in a pre-processing step, we use a lower bound L on the optimal objective obtained from heuristics for the 2cPCKP. See Appendix A for the heuristics we use. Moreover, we exploit precedence relations among items, similar as in the deterministic PCKP case (see Boland et al. (2012) and You & Yamada (2007)). The difference from the deterministic setting, however, is that we also incorporate the uncertainty in item weights and intensively use the following lower and upper bounds on the cardinality of the optimal solutions.

Bounds on the Cardinality of Optimal Solutions: To present our bounds on the cardinality of optimal solutions, we first introduce some notation. We let $\overline{p}(n, I)$ and $\underline{p}(n, I)$ denote the maximum and minimum profit of n items from the set I, respectively. That is,

$$\overline{p}(n, I) = \max_x \Big\{ \sum_{i \in I} p_i x_i : \sum_{i \in I} x_i = n, x \in \{0,1\}^{|I|} \Big\}, \tag{15}$$

and

$$\underline{p}(n, I) = \min_x \Big\{ \sum_{i \in I} p_i x_i : \sum_{i \in I} x_i = n, x \in \{0,1\}^{|I|} \Big\}. \tag{16}$$

Similarly, $\overline{w}(n, I)$ and $\underline{w}(n, I)$ represent the maximum and minimum deterministic weight of n items in set I, respectively, $\overline{\mu}(n, I)$ and $\underline{\mu}(n, I)$ represent the maximum and minimum uncertain weight means of n items in set I, respectively, and $\overline{\sigma}^2(n, I)$ and $\underline{\sigma}^2(n, I)$ represent the maximum and minimum uncertain weight variances of n items in set I, respectively.

All these values can be easily computed by sorting the vector p (or w, μ, or σ^2) either in ascending or descending order and summing over the first n items. Based on these values we derive the following bounds on the number of items selected, $\sum_i x_i^*$, in an optimal solution (x^*, y^*) to the 2cPCKP:

$$n^l = \max\{n : \overline{p}(n, I) < L\}, \tag{17}$$

$$n^u = \min\{n : \underline{w}(n, I) + \underline{\mu}(\lceil Rn \rceil, I) + \Phi^{-1}(\alpha)\sqrt{\underline{\sigma}^2(\lceil Rn \rceil, I)} > B\}. \tag{18}$$

Here, n^l represents a lower bound on $\sum_i x_i^*$ since in an optimal solution the profit should not be lower than the lower bound L. Similarly, n^u is an upper bound on $\sum_i x_i^*$ since (18) shows that any set of items with cardinality n^u and above will violate the capacity constraint of the knapsack. Here, we use that if $\sum_{i \in I} x_i^* = n$, then $\sum_{i \in I} y_i^* \geq \lceil Rn \rceil$. We conclude that for any optimal solution (x^*, y^*), the following cardinality constraint holds.

$$n^l \leq \sum_{i \in I} x_i^* \leq n^u \tag{19}$$

Eliminating Items Based on Cardinality Bounds and Precedence Constraints: The precedence constraints imply that if item $i \in I$ is added to the knapsack, then all predecessors $P(i)$ of item i also need to be selected. If there are too many of such predecessors, i.e., if there are more than n^u, then we know that item i is not selected in

any optimal solution. Hence, we can eliminate item i if,

$$|P(i)| + 1 > n^u. \tag{20}$$

However, if this inequality does not hold, i.e., $|P(i)| + 1 \leq n^u$, then there are still two reasons why we may eliminate item i. First, the profits corresponding to item i and its predecessors may be too low, namely,

$$p_i + \sum_{j \in P(i)} p_j + \overline{p}\big(n^u - (|P(i)| + 1), I - (P(i) \cup \{i\})\big) < L. \tag{21}$$

In this case, the profits will not reach the lower bound L when at most n^u items are selected with item i and its predecessors $P(i)$ included. Second, we can eliminate item i if the weight of item i and its predecessors is too large, namely,

$$w_i + \sum_{j \in P(i)} w_j + \underline{w}\big(n^l - (|P(i)| + 1), I - (P(i) \cup \{i\})\big) > B. \tag{22}$$

In this case, the capacity B will be insufficient when at least n^l items are selected with item i and its predecessors $P(i)$ included. Note that with $\underline{w}\big(n^l - (|P(i)| + 1), I - (P(i) \cup \{i\})\big)$ the equation (22) strengthens the standard weight-based elimination procedure used for PCKP, as the procedure used for PCKP eliminates item i when $w_i + \sum_{j \in P(i)} w_j > B$.

4.2 Cutting Plane Methods

The deterministic formulation of the 2cPCKP is nonlinear for $\alpha > 0.5$, due to the term $\sqrt{\sum_{i \in I} \sigma_i^2 y_i^2}$ in constraint (10). This nonlinearity poses a challenge for the state-of-the-art IP solvers, already for reasonably sized problem instances. To overcome the nonlinearity, we propose cutting plane methods that linearize $\sqrt{\sum_{i \in I} \sigma_i^2 y_i^2}$ and refine the feasibility space iteratively by adding feasibility cuts.

4.2.1 Exact Method

Our exact cutting plane method is based on the following theoretical result.

Lemma 4.1. *Consider the function $g : \mathbb{R}_+^{|I|} \to \mathbb{R}$ defined for every $y \in \mathbb{R}_+^{|I|}$ as $g(y) = \sqrt{\sum_{i \in I} \sigma_i^2 y_i^2}$. Then,*

(i) the function g is a convex *function of y, and*

(ii) for every fixed point $\hat{y} \in \mathbb{R}_+^{|I|}$ with $\hat{y} \neq 0$, it holds that $g(y) \geq g(\hat{y}) + \nabla g(\hat{y})^T (y - \hat{y})$ for all $y \in \mathbb{R}_+^{|I|}$, where $\nabla g(\hat{y})^T (y - \hat{y}) = \frac{1}{\sqrt{\sum_{i \in I} \sigma_i^2 \hat{y}_i^2}} \sum_{i \in I} \sigma_i^2 \hat{y}_i (y_i - \hat{y}_i)$.

Proof. The convexity of $g(y)$ follows from the fact that g is a weighted Euclidean norm. Since g is convex, every first-order linear approximation of g is a lower bound; see, e.g., Section 3.1 in S. Boyd & Vandenberghe (2004). □

Since g is convex, we can represent g as the supremum over its tangent planes $g(\hat{y}) + \nabla g(\hat{y})^T (y - \hat{y})$. Moreover, since $y \in \{0,1\}^{|I|}$ is binary in the 2cPCKP, we only need finitely many tangent planes to get an exact dual representation of g at $\{0,1\}^{|I|}$. However, instead of using all tangent planes directly, we add them in an iterative scheme. In iteration k of this cutting plane method, we assume that there are k tangent planes available and we define the approximate feasible region $Z^k := \{(x,y) | \sum_i w_i x_i + \sum_i \mu_i y_i + \Phi^{-1}(\alpha)(g(\hat{y}) + \nabla g(\hat{y}^l)^T (y - \hat{y}^l)) \leq B, l = 1, 2, .., k,$ *and satisfying* $(11) - (14)\}$, and solve $P^k := \max_{x,y} \{\sum_i p_i x_i | (x,y) \in Z^k\}$, yielding current

optimal solution (x^k, y^k). Note that it follows from Lemma 4.1 that $Z^k \supset Z$, with Z the real feasible region of the 2cPCKP. Hence, if the current solution $(x^k, y^k) \in Z$, it is not only optimal for the approximate problem P^k, but also for the 2cPCKP. If $(x^k, y^k) \notin Z$, then we add a tangent plane to Z^k to make sure that (x^k, y^k) is infeasible in the next iteration. Algorithm 1 describes our cutting plane procedure. It is given in a generic form for an arbitrary function G, since later we will use this same algorithm approximately using approximate functions for g.

Algorithm 1 Pseudocode of the cutting plane methods

1: Input: $w_i, \mu_i, \sigma_i^2, p_i, \forall i \in I$, α, B and a function $G(y)$;
2: Initialize $k = 0$ and set $Z^0 = \{(x, y) | \sum_i w_i x_i + \sum_i \mu_i y_i \leq B \text{ and satisfying } (11) - (14)\}$;
3: Solve $P^k = \max_{x,y}\{\sum_i p_i x_i | (x, y) \in Z^k\}$ and let $(x^k, y^k) \in Z^k$ be an optimal solution to P^k;
4: **if** $(x^k, y^k) \notin Z$ **then**
5: Add a tangent plane to Z^k:
6: Let $Z^{k+1} = Z^k \cap \{(x, y) | \sum_i w_i x_i + \sum_i \mu_i y_i + \Phi^{-1}(\alpha)(G(y^k) + \nabla G(y^k)^T (y - y^k)) \leq B\}$;
7: Set $k \leftarrow k + 1$ and go to line 3;
8: **else**
9: **return** (x^k, y^k) with profit $\sum_i p_i x_i$;
10: **end if**

Our exact cutting plane method uses $G(y) = g(y) = \sqrt{\sum_{i \in I} \sigma_i^2 y_i^2}$ in Algorithm 1. In the first iteration of the algorithm, we solve the nominal 2cPCKP with the capacity constraint $\sum_i w_i x_i + \sum_i \mu_i y_i \leq B$. This implies that if $y^0 = 0$, then $(x^0, y^0) \in Z$, and is thus also optimal for the 2cPCKP. Typically, however, $y^0 \neq 0$ and $(x^0, y^0) \notin Z$, which implies that we use the first-order Taylor approximation of g at y^0, given by

$$g(y^0) + \nabla g(y^0)^T (y - y^0) = \sqrt{\sum_{i \in I} \sigma_i^2 {y_i^0}^2} + \frac{1}{\sqrt{\sum_{i \in I} \sigma_i^2 {y_i^0}^2}} \sum_{i \in I} \sigma_i^2 y_i^0 (y_i - y_i^0), \tag{23}$$

to refine the approximate feasible region Z^0 to Z^1. The algorithm terminates in finitely many steps since in total only finitely many tangent planes can be added. We must note that in each iteration problem P^k is an integer linear program corresponding to a nominal 2cPCKP with k deterministic linear capacity constraints.

4.2.2 Approximate Methods

The use of the exact cutting plane method can be computationally burdensome; even if the solutions to the linear problems P^k in Algorithm 1 can be found efficiently, the method may take quite a number of iterations until it converges. This may not be a problem if we would have quickly obtained sub-optimal solutions with a small optimality gap. However, in this case this is a potential problem since every solution (x^k, y^k) to P^k that we find before the final iteration is *infeasible* for the original 2cPCKP problem. Therefore, in order to supplement our exact cutting plane method, we propose approximate cutting plane methods that quickly yield feasible solutions. We do so by using Algorithm 1 with different functions for G.

In the first approximate cutting plane method, we use $G(y) = \tilde{g}(y) = \sqrt{\sum_{i \in I} \sigma_i^2 y_i}$, which has the following properties.

Lemma 4.2. *Consider the function* $\tilde{g} : \mathbb{R}_+^{|I|} \to \mathbb{R}$ *defined for every* $y \in \mathbb{R}_+^{|I|}$ *as* $\tilde{g} = \sqrt{\sum_{i \in I} \sigma_i^2 y_i}$. *Then,*

(i) the function $\tilde{g}$ *is a* concave *function of* y, *and*

(ii) for every fixed point $\hat{y} \in \mathbb{R}_+^{|I|}$ *with* $\hat{y} \neq 0$*, it holds that* $\tilde{g}(y) \leq \tilde{g}(\hat{y}) + \nabla\tilde{g}(\hat{y})^T(y - \hat{y})$ *for all* $y \in \mathbb{R}_+^{|I|}$*, where* $\nabla\tilde{g}(\hat{y})^T(y - \hat{y}) = \frac{1}{2\sqrt{\sum_{i\in I}\sigma_i^2\hat{y}_i}} \sum_{i\in I}\sigma_i^2(y_i - \hat{y}_i)$.

Proof. The concavity of $\tilde{g}(y)$ follows from the concavity of the square root function. Using the result that $-\tilde{g}(y)$ is a convex function and for convex functions their first-order linear approximations provide lower bounds, we infer the relation in (ii). □

Note that $\tilde{g}(\hat{y}) = g(\hat{y})$ for any given $\hat{y} \in \{0,1\}^N$. This implies that the approximation $\tilde{g}(\hat{y}) + \nabla\tilde{g}(\hat{y})^T(y - \hat{y})$ at $\hat{y}$ will also give an upper bound for $g(y)$ at $y \in \{0,1\}^N$. In the following, we formally define the approximate method which we refer to as the "2-iteration method" in the rest of the paper.

Definition 4.1. ***"2-iteration method"****: Use the cutting plane method described in Algorithm 1 with* $G(y) = \tilde{g}(y) = \sqrt{\sum_{i\in I}\sigma_i^2 y_i}$.

In the following, we show that the 2-iteration method converges to feasible solutions within two iterations.

Proposition 4.1. *The 2-iteration method provides a feasible solution for the 2cPCKP in at most two iterations.*

Proof. Suppose that the solution (x^0, y^0) to P^0 in the first iteration of the 2-iteration method is infeasible for the 2cPCKP. Then, we add the constraint

$$\sum_i w_i x_i + \sum_i \mu_i y_i + \tilde{g}(y^0) + \nabla\tilde{g}(y^0)^T(y - y^0) \leq B \tag{24}$$

to Z^1. However, since $g(y) = \tilde{g}(y) \leq \tilde{g}(y^0) + \nabla\tilde{g}(y^0)^T(y - y^0)$ for all $y \in \{0,1\}^{|I|}$, it follows that $Z^1 \subset Z$, and thus the optimal solution to P^1 will also be feasible for the 2cPCKP. Hence, the algorithm terminates in at most two iterations. □

Since the 2-iteration method terminates in at most two iterations, its convergence will be much faster than the exact cutting plane method. The disadvantage of the 2-iteration method over the exact method, however, is that the 2-iteration method may overestimate $g(y)$, yielding a positive optimality gap. We propose to combine both approaches, exploiting the advantages of both. We refer to these approximate cutting plane methods as *convex-concave mixture methods.*

Definition 4.2. ***"Convex-concave mixture methods"****: Let* $\theta \in [0,1]$ *and use the cutting plane method described in Algorithm 1 with* $G(y) = \hat{g}_\theta(y) = \sqrt{\sum_{i\in I}\sigma_i^2(\theta y_i + (1-\theta)y_i^2)}$*. For any given* $y \in \mathbb{R}_+^{|I|}$ *and* $\hat{y} \in \mathbb{R}_+^{|I|}$*,* $\nabla\hat{g}_\theta(y)^T(y - \hat{y}) = \frac{1}{2\sqrt{\sum_{i\in I}\sigma_i^2(\theta y_i + (1-\theta)y_i^2)}} \sum_{i\in I}\sigma_i^2(\theta + 2(1-\theta)\hat{y}_i)(y_i - \hat{y}_i)$.

For $\theta = 0$ the method above reduces to the exact cutting plane method since $\hat{g}_\theta(y) = g(y)$, and for $\theta = 1$ it reduces to the 2-iteration method since $\hat{g}_\theta(y) = \tilde{g}(y)$. In general, however, there are infinitely many approximate cutting plane methods of this type, one for each $\theta \in [0,1]$, where θ can be interpreted as a weight parameter. The closer θ gets to 0, the closer the method will resemble the exact cutting plane method, whereas the closer θ gets to 1, the closer it will resemble the 2-iteration method. We focus particularly on the following equal-mix ($\theta = 0.5$) method, which we refer to as the "midway method".

Definition 4.3. ***"Midway method"****: Use the cutting plane method described in Algorithm 1 with* $G(y) = \hat{g}(y) = \sqrt{\sum_{i\in I}\sigma_i^2\frac{(y_i + y_i^2)}{2}}$.

Although we expect the midway method to find solutions with better objective values than the 2-iteration method, it is not completely clear whether this always has to be the case. However, in Proposition 4.2 below, we provide conditions under which the quality of the midway method solution is guaranteed to be better than that of the 2-iteration method.

Proposition 4.2. *Let $Z^k(2)$ and $Z^k(m)$ denote the approximate feasible regions of P^k in iteration k of the 2-iteration and midway methods, respectively. Also, let $y^k(2)$ and $y^k(m)$ denote optimal y solutions for P^k of the 2-iteration and midway methods, respectively. Then,*

(i) $Z^1(2) \subseteq Z^1(m)$, and

(ii) $Z^1(2) \subseteq Z^k(m)$, if $\sum_{i\in I} \sigma_i^2 y_i^0(2) \geq \sum_{i\in I} \sigma_i^2 y_i^l(m)$ holds for all $l \in \{0, 1, ..., k\}$.

Proof. We first note that $y^0(2) = y^0(m)$. Therefore, let us denote this as simply y^0. To prove that (i) holds, it suffices to show that

$$\tilde{g}(y^0) + \nabla\tilde{g}(y^0)^T(y - y^0) \geq \hat{g}(y^0) + \nabla\hat{g}(y^0)^T(y - y^0), \forall y \in \{0,1\}^{|I|}. \tag{25}$$

Since $y_i = \frac{y_i + y_i^2}{2}$ for any $y_i \in \{0, 1\}$, $\tilde{g}(y^0) = \hat{g}(y^0)$ for any $y^0 \in \{0, 1\}^{|I|}$. We then focus on showing that $\nabla\tilde{g}(y^0)^T(y - y^0) \geq \nabla\hat{g}(y^0)^T(y - y^0)$ holds. Let us write this inequality in its extended form, which is

$$\frac{1}{2\sqrt{\sum_i \sigma_i^2 y_i^0}} \sum_i \sigma_i^2(y_i - y_i^0) \geq \frac{1}{2\sqrt{\sum_i \sigma_i^2 \frac{y_i^0 + y_i^{0^2}}{2}}} \sum_i \sigma_i^2(1/2 + y_i^0)(y_i - y_i^0). \tag{26}$$

By using $\tilde{g}(y^0) = \hat{g}(y^0)$ again, we know that the above inequality holds when

$$\sum_i \sigma_i^2(y_i - y_i^0) \geq \sum_i \sigma_i^2(1/2 + y_i^0)(y_i - y_i^0). \tag{27}$$

We then verify that this inequality holds by checking that $y_i - y_i^0 \geq (1/2 + y_i^0)(y_i - y_i^0)$ for any $y_i, y_i^0 \in \{0, 1\}$. Note that we need to evaluate four cases: (a) $y_i = 0$ and $y_i^0 = 0$, (b) $y_i = 0$ and $y_i^0 = 1$, (c) $y_i = 1$ and $y_i^0 = 0$ and (d) $y_i = 1$ and $y_i^0 = 1$. Indeed, simple computations show that this condition holds in all four cases.

We now prove (ii). Let us denote $\sum_i \sigma_i^2 y_i^0$ by A_0 and $\sum_i \sigma_i^2 y_i^l(m)$ by A_l. Let (x, y) be an arbitrary element of $Z^1(2)$. Then, (x, y) satisfies

$$\sum_i w_i x_i + \sum_i \mu_i y_i \leq B - \Phi^{-1}(\alpha)\frac{\sqrt{A_0}}{2} - \sum_{i\in I} \frac{\Phi^{-1}(\alpha)\sigma_i^2}{2\sqrt{A_0}} y_i, \tag{28}$$

which is derived from $\sum_i w_i x_i + \sum_i \mu_i y_i + \tilde{g}(y^0) + \nabla\tilde{g}(y^0)^T(y - y^0) \leq B$. In $P^k(m)$, there are k capacity constraints. We show that (x, y) satisfies an arbitrary selected lth capacity constraint resulting from the solution $(x^l(m), y^l(m))$, if $A_0 \geq A_l$. For this to hold, it must be that

$$\sum_i w_i x_i + \sum_i \mu_i y_i + \sum_{i\in I} \frac{\Phi^{-1}(\alpha)\sigma_i^2}{2\sqrt{A_l}}\left(\frac{y_i - y_i^l}{2} + y_i y_i^l\right) \leq B - \Phi^{-1}(\alpha)\frac{\sqrt{A_l}}{2}, \tag{29}$$

which we derive from $\sum_i w_i x_i + \sum_i \mu_i y_i + \hat{g}(y^l(m)) + \nabla\hat{g}(y^l(m))^T(y - y^l(m)) \leq B$. Using (28) to bound $\sum_i w_i x_i + \sum_i \mu_i y_i$ from above, for showing (29), it suffices to show that

$$\sum_{i\in I} \frac{\sigma_i^2}{\sqrt{A_l}}\left(\frac{y_i - y_i^l}{2} + y_i y_i^l - \frac{y_i}{\sqrt{A_1}}\right) \leq \sqrt{A_1} - \sqrt{A_l} \tag{30}$$

holds. Given that $\sqrt{A_1} - \sqrt{A_l} \geq 0$ and $\sum_{i\in I} \frac{\sigma_i^2}{\sqrt{A_l}}\left(\frac{y_i - y_i^l}{2} + y_i y_i^l - \frac{y_i}{\sqrt{A_1}}\right) \leq 0$, we can confirm that (30) holds. To see that $\sum_{i\in I} \frac{\sigma_i^2}{\sqrt{A_l}}\left(\frac{y_i - y_i^l}{2} + y_i y_i^l - \frac{y_i}{\sqrt{A_1}}\right) \leq 0$, for arbitrary binary vectors y, y^l, we consider all the possible cases that y_i and y_i^l can take for any item i. If $y = y^l = 0^{|I|}$, then $\sum_{i\in I} \frac{\sigma_i^2}{\sqrt{A_l}}\left(\frac{y_i - y_i^l}{2} + y_i y_i^l - \frac{y_i}{\sqrt{A_1}}\right) = 0$. However, if there exists item i with $y_i = 0, y_i^l = 1$, or $y_i = 1, y_i^l = 1$, or $y_i = 1, y_i^l = 0$, then $\sum_{i\in I} \frac{\sigma_i^2}{\sqrt{A_l}}\left(\frac{y_i - y_i^l}{2} + y_i y_i^l - \frac{y_i}{\sqrt{A_1}}\right) < 0$. □

We note that the condition given in Proposition 4.2 is very likely to hold for many problem instances. Given that P^0 solves the nominal 2cPCKP where the variances of uncertain item weights are ignored completely, whereas in $P^k, k \geq 1$ the variances are incorporated through the feasibility cuts, the sum of the variances of the items selected with the non-preferred method ($y_i^0 = 1$) in P^0 is likely to be higher than of $P^k, k \geq 1$.

Recall that the solution (x^0, y^0) obtained for P^0 with our cutting plane methods (in both exact and approximate methods) correspond to the optimal solution of the nominal 2cPCKP in which the variances of the uncertain item weights are ignored and only the deterministic weights and the means of the uncertain item weights are considered in the capacity constraint. The 2-iteration method uses this nominal optimal solution to produce the Taylor approximation $\tilde{g}(y^0) + \nabla\tilde{g}(y^0)^T(y - y^0)$, which overestimates $g(y)$, at the second iteration, namely, its final iteration. The extent of this overestimation may play an important role in the solution quality that the 2-iteration method produces, as a higher overestimation used for $g(y)$ in the capacity constraint can lead to using the knapsack capacity more conservative than necessary in solving P^1. In the following, we provide bounds for this over estimation (i.e., $\tilde{g}(y^0) + \nabla\tilde{g}(y^0)^T(\tilde{y} - y^0) - g(\tilde{y})$, for any given $\tilde{y} \in \{0,1\}^N$) in terms of the nominal solution y^0.

Proposition 4.3. *Let (x^0, y^0) be the solution to P^0 at the first iteration of the 2-iteration method, also let $U = \{i \in I : y_i^0 = 1\}$ and $I - U = \{i \in I : y_i^0 = 0\}$. Then, for any given $y \in \{0,1\}^N$, $\tilde{g}(y^0) + \nabla\tilde{g}(y^0)^T(y - y^0) - g(y)$ will be at most $(\sqrt{\sum_{i\in U}\sigma_i^2})/2$ when $\sum_{i\in U}\sigma_i^2 > \frac{1}{3}\sum_{i\in I-U}\sigma_i^2$, otherwise it will be at most $\sqrt{\sum_{i\in U}\sigma_i^2} + \frac{\sum_{i\in I-U}\sigma_i^2}{2\sqrt{\sum_{i\in U}\sigma_i^2}} - \sqrt{\sum_{i\in U}\sigma_i^2 + \sum_{i\in I-U}\sigma_i^2}$.*

Proof. The estimation gap $\tilde{g}(y^0) + \nabla\tilde{g}(y^0)^T(y - y^0) - g(y)$ is equal to

$$= \sqrt{\sum_{i\in U}\sigma_i^2} + \sum_i \frac{\sigma_i^2}{2\sqrt{\sum_{i\in U}\sigma_i^2}}(y_i - y_i^0) - \sqrt{\sum_i \sigma_i^2 y_i} \tag{31}$$

$$= \sqrt{\sum_{i\in U}\sigma_i^2} + \frac{\sum_{i\in U}\sigma_i^2(y_i - 1) + \sum_{i\in I-U}\sigma_i^2 y_i}{2\sqrt{\sum_{i\in U}\sigma_i^2}} - \sqrt{\sum_{i\in U}\sigma_i^2 y_i + \sum_{i\in I-U}\sigma_i^2 y_i}. \tag{32}$$

Firstly, we note that the gap is the smallest, equal to zero, for $y = y^0$, i.e., $y_i = 1, \forall i \in U, y_i = 0, \forall i \in I - U$. However, the estimation gap can be positive as y differentiates from y^0. The most differentiated y for y^0 is y which lets $y_i = 0, \forall i \in U$ and $y_i = 1, \forall i \in I - U$. With this y, the gap becomes

$$\frac{1}{2}\sqrt{\sum_{i\in U}\sigma_i^2} + \frac{\sum_{i\in I-U}\sigma_i^2}{2\sqrt{\sum_{i\in U}\sigma_i^2}} - \sqrt{\sum_{i\in I-U}\sigma_i^2} \tag{33}$$

However, this gap may not be the largest gap possible. Now, let us consider some other y which again lets $y_i = 1, \forall i \in I - U$, however, lets $y_i = 1$ also for some $i \in U$. If the gap with such a y has the potential to be larger than (33), it will attain its largest with y which lets $y_i = 1, \forall i \in I - U$ and $y_i = 1, \forall i \in U$. The gap under such a y is equal to

$$\sqrt{\sum_{i\in U}\sigma_i^2} + \frac{\sum_{i\in I-U}\sigma_i^2}{2\sqrt{\sum_{i\in U}\sigma_i^2}} - \sqrt{\sum_{i\in U}\sigma_i^2 + \sum_{i\in I-U}\sigma_i^2}. \tag{34}$$

With algebraic manipulations we can derive that (33) will be larger than (34) only when $\sum_{i\in I-U}\sigma_i^2 < \frac{3}{2}\sum_{i\in U}\sigma_i^2$. Similarly, we consider some other y which lets $y_i = 0, \forall i \in U$, however, lets $y_i = 0$ also for some $i \in I - U$. Similarly, to explore the largest possible gap under such a y, we consider the gap with y which lets $y_i = 0, \forall i \in U$ and $y_i = 0, \forall i \in I - U$. With this y, the gap is $\frac{1}{2}\sqrt{\sum_{i\in U}\sigma_i^2}$. We can derive that (33) will be larger than this gap only when $\sum_{i\in I-U}\sigma_i^2 > 4\sum_{i\in U}\sigma_i^2$. Since it will never happen that this condition and

the condition above ($\sum_{i\in I-U}\sigma_i^2 < \frac{3}{2}\sum_{i\in U}\sigma_i^2$) are satisfied together, for considering the largest gap, we can disregard (33), and focus on (34), and its relation to $\frac{1}{2}\sqrt{\sum_{i\in U}\sigma_i^2}$. With algebraic manipulations we can derive that $\frac{1}{2}\sqrt{\sum_{i\in U}\sigma_i^2}$ will be larger than (34) as long as $\sum_{i\in U}\sigma_i^2 > \frac{1}{3}\sum_{i\in I-U}\sigma_i^2$. □

The solution for y at the first iteration y^0 can prefer selecting items whose uncertain weights have small means. When there is a negative correlation between the means and variances of the uncertain item weights such that items with small uncertain weight means have very large variances, and vice versa, the sum $(\sqrt{\sum_{i\in U}\sigma_i^2})/2$, where $U = \{i \in I : y_i^0 = 1\}$, can be very high and many of the items selected in y^0 may not be selected in an optimal solution for y. We foresee that this can cause significant optimality gaps in the solutions produced by the 2-iteration method. We provide numerical evidence for this in Section 5.2.

4.3 Cover Cuts

We propose new cover cuts for the 2cPCKP, for improving the computational efficiency of the exact cutting plane method which may take many iterations.

A cover $C \subseteq I$ is a set of items which together do not fit into the knapsack. More specifically, if C is a cover, then

$$\sum_{i\in C} x_i \leq |C| - 1 \tag{35}$$

is a valid inequality, which we refer to as a cover cut.

Cover cuts proposed for the deterministic PCKP (see E. A. Boyd (1993) and Van de Leensel et al. (1999)) integrate the precedence constraints in confirming that a given subset $C \subseteq I$ of items will not fit into the knapsack. Let $P(C) = \bigcup_{i\in C} P(i) \cup C$ denote the minimal set of items selected in the knapsack so that it is possible to select all items in C without violating any precedence constraints. This means that in the deterministic case the capacity B should be at least $\sum_{i\in P(C)} w_i$, or otherwise C is a cover. In this paper, we extend this approach to the 2cPCKP by incorporating the uncertain item weights. In achieving this, different from the cover cuts proposed for the deterministic PCKP, we utilize a lower bound L on the optimal objective value intensively. The procedure we use in checking if a given set of items C is a cover is described as follows.

Declaring Covers. Firstly, similar to the pre-processing procedure which we use in eliminating items, we use the upper cardinality bound n^u. We declare C a cover if

$$|P(C)| > n^u. \tag{36}$$

Secondly, we focus on the item weights. We extend the condition $\sum_{i\in P(C)} w_i > B$ by using a lower bound on the number of items that will be selected from $I - P(C)$ in a solution that selects all the items in $P(C)$. We denote this bound with n^o and find it by

$$n^o = \max\{n : \overline{p}(n, I - P(C)) \leq L - \sum_{j\in P(C)} p_j\}. \tag{37}$$

Note that n^o can be found very efficiently by simple binary search. Then, based on the minimum weight of items in a solution that selects the items in C, and comparing this weight to the available knapsack capacity B, we declare C a cover if

$$\sum_{i\in P(C)} w_i + \underline{w}(n^o, I - P(C)) + \underline{\mu}(\lceil R(n^o + |P(C)|)\rceil, I) + \Phi^{-1}(\alpha)\sqrt{\underline{\sigma}^2(\lceil R(n^o + |P(C)|)\rceil, I)} > B. \tag{38}$$

Note that in (38) we consider only the items in $I-P(C)$ for calculating $\underline{w}(n^o, I-P(C))$, while for $\underline{\mu}(\lceil R(n^o+|P(C)|)\rceil, I)$ and $\underline{\sigma}^2(\lceil R(n^o+|P(C)|)\rceil, I)$ we consider the item set I. The reason for this is that the decisions on which items will be selected by the non-preferred method in solving the 2cPCKP depends on the entire solution (see 11).

The Procedure Used for Finding Cover Cuts: During the search process of the branch and bound of the IP solvers, at a given node of the search tree, we use the relaxation solutions x^* of x at the node for adding cover cuts. In generating these, we use a random constructive heuristic which attempts to find a set of items C which can be declared as a cover by (36) or (38). The procedure attempts to find a cover C with $\sum_{i\in C} x_i^* > |C|-1$.

Initially, the heuristic starts with an empty selection $C=\emptyset$ and at each iteration it attempts to add a randomly selected new item to C, for a limited number of trials, unless C is declared to be a cover already by the above procedure. The added new item at an iteration is checked for incomparability (see Van de Leensel et al. (1999) for the definition of incomparability); it should not be a predecessor, or a successor, of the items that are in the current selection.

5 Computational Experiments

In this section, we investigate the performance of the proposed methods for solving the 2cPCKP. The methods are implemented in C++ using an Intel Xeon 2.5 GHz processor with 128 GB memory. To solve the ILP models at every iteration of the cutting plane methods, we use Gurobi 9.1 with optimality gap tolerance of 0.01% and a time limit of one hour. First, however, we use the heuristics proposed in Appendix A, i.e., the topological sorting and randomized construction heuristics, to find a lower bound L on the optimal objective value. This lower bound L is equal to the objective value of the best heuristic solution that we find, and we use this lower bound in our pre-processing procedures of Section 4.1 and cover cuts of Section 4.3. When using the randomized construction heuristic, we limit the number of iterations to the number of items $|I|$, while we fix the limit on the number of non-improving iterations at 100. We note that the topological sorting heuristic uses a topological ordering of items I in the graph $G=(I,E)$. In finding this, we use a depth-first search approach.

To investigate the performance of our methods, we use artificially generated problem instances with parameter settings as described in Table 1. We perform a full factorial design approach over the considered parameter settings. When generating our problem instances, some parameters are drawn from a probability distribution. To account for this randomness, we take 10 samples per parameter setting, yielding 10 problem instances, and we present the average results over these 10 instances.

In the experiments, given the number of items $|I|$, we generate directed acyclic precedence graphs $G=(I,E)$ by controlling two parameters: the breadth b and density d of the graph. A graph G with breadth b has $|I|$ vertices on $|I|/b$ levels, each level consisting of exactly b vertices. The vertices on the first level represent items that do not have predecessors, and have at least one immediate successor among the items on the second level. We assume that there are only directed edges between the vertices on level k to the vertices on level $k+1, k=1,...,|I|/b-1$. The density d determines how many edges exist in the network: for each pair of vertices between consecutive levels we generate an edge with probability d/b so that on average there are d edges from a vertex in level k to the vertices in level $k+1$.

The values we set for B, w and p are similar to the ones experimented in You & Yamada (2007) for the deterministic PCKP. Given that on average the deterministic weight of an item will be around 500, by letting

Table 1: Parameter settings of the instances.

Parameters	Values	Explanation
$\|I\|$	10, 50, 100, 1000, 2000, 4000, 8000	Number of items
b	50, 100, 200 if $\|I\| \geq 1000$, otherwise $b = 5$	Breadth of the precedence graph
d	1, 5, 10	Density of the precedence graph
R	0.5	Uncertainty rate
α	0.95	Confidence level
B	$250\|I\|$	Knapsack capacity
w_i	$\sim U[1, 1000]$	Deterministic weight of item i
$\xi_i - w_i$	$\sim N(\mu_i, \sigma_i^2)$	Uncertain weight of item i where $\mu_i \sim N(250, 200^2)$ and $\sigma_i \sim U[1, 200]$
p_i	UC, WC, SC	Profit of item i where UC lets $p_i \sim U[1, 1000]$ WC lets $p_i \sim U[w_i, w_i + 200]$ SC lets $p_i = w_i + 200$

$B = 250|I|$ we expect that at most a half of the items can be placed in the knapsack. Item profits are sampled in three different ways, to model that the profits and deterministic item weights may be uncorrelated (UC), weakly correlated (WC), and strongly correlated (SC).

Again similar to You & Yamada (2007), we consider cases for $|I|$ with $|I| \geq 1000$. Note that the use of systematic optimization methods such as IP solvers is reasonable for knapsack problems only when the number of items available to select is significantly large because, otherwise, optimal solutions can be detected instead by using simpler approaches such as enumeration and evaluation. Nevertheless, in Table 1, we also include small instances with $|I| \leq 100$. This way we can better detect the optimality gaps of the approximate methods. Note that as the number of items available ($|I|$) in instances gets larger, the effect by the variances of uncertain item weights ($\sigma_i^2 s$) in the knapsack capacity will be a lot smaller than by the means of the uncertain item weights ($\mu_i s$). For example, if $\sigma_i = \sigma, \forall i$ and $\mu_i = \mu, \forall i$, then $\sqrt{|I|\sigma^2}$ will be much smaller than $|I|\mu$ as $|I|$ increases. When this is the case, the solutions obtained by the nominal 2cPCKP which disregards σ_i^2s may not be too different from the optimal solutions, and our approximate cutting plane methods that use the nominal solution at the first iteration can produce high-quality feasible solutions. In the rest of the paper, we refer to these instances with $|I| \leq 100$ as small instances and the other instances with $|I| \geq 1000$ as large instances.

Our settings on the parameter d, where the smallest value considered is 1, reflect the difficulty of real-life PCKP instances that involve generally high-density precedence graphs in which the number of edges is at least as large as the number of items. For example, the precedence graphs of real-life telecommunication and open-pit mine production scheduling problems (see the instances in Boland et al. (2012)) often have a density level of above 1 and can reach a density of nearly 10. We note that precedence graphs in personalized learning relating to the prerequisite relations among subjects can be highly dense as well. As an example of this, we refer to the prerequisite graphs of the courses (named curriculum map) offered at MIT (see `https://rhumbl.com/examples/curriculum-maps`).

We measure the optimality gap of an approximate method on an instance as follows. Let z^* be the optimal objective value of the instance and z be the objective value of a solution to the instance by an approximate method. The optimality gap is equal to $100 \times (z^* - z)/z$.

In Section 5.1, we present the results on the performance of our pre-processing procedures. Section 5.2 investigates the performance of our approximate cutting plane methods. In this section, we also investigate the computational performance of Gurobi 9.1 in solving the quadratic formulation (9)-(14). In Section 5.3, the effect

of cover cuts is investigated in improving the computational efficiency of the exact cutting plane method. Section 5.4 illustrates a numerical example based on a real-life situation in a university setting where the 2cPCKP can be used. In Section 5.5, we benchmark the 2-iteration method in solving the the chance-constrained binary knapsack problem with independent and normally distributed weights (Han et al., 2016; Joung & Lee, 2020), which is a special case of our problem. The performance analysis on the heuristics used for obtaining lower bounds for the 2cPCKP is provided in Appendix B.

5.1 Performance of the Pre-Processing Procedures

In Table 2, we present the item elimination performance of the pre-processing procedures on the large instances, where we can see the effect of p_i, b, d and $|I|$ on the average number of items eliminated with the pre-processing procedures. We notice that the most important factor affecting the ability of pre-processing procedures to eliminate items is the density of precedence graphs. It seems that when $d = 1$, pre-processing is not able to eliminate many items, however, when $d = 5$ or $d = 10$, it eliminates nearly up to a half of the items. As the precedence relations among items increase in density, placing some item i in the knapsack will require many other items to be also included in the knapsack. This will mean that the pre-processing procedure will find higher lower bounds on the total capacity required in a solution that lets $x_i = 1$. With higher bounds, it will be more likely to exceed the knapsack capacity B and thus to eliminate specific items. In Table 2, we also identify the trend that as the depth of precedence graphs increases (i.e., when b decreases), pre-processing becomes more effective. This can be also related to the effect that the set of items that lie on the path to reach an item may get large as the precedence graphs increase in depth. Another observation is that different correlation settings used for modeling item profits do not affect the effectiveness of pre-processing to a significant degree. The last two rows of Table 2 show the overall average number of items eliminated in the instances with $|I| = 1000$, $|I| = 2000$, $|I| = 4000$ and $|I| = 8000$, averaged over all settings for p_i, b and d, and the percentages of these with respect to $|I|$. These averages show that as the number of items increases, more than 30% of the items can be eliminated. We conclude that pre-processing procedures can become very effective in the presence of high-density precedence relations among items and this effectiveness becomes more eminent in large instances.

5.2 Performance of the Approximate Cutting Plane Methods

Let us firstly focus on the performance of our approximate cutting plane methods in finding high quality feasible solutions, namely on their optimality gaps in the small and large instances. In Table 3, we present the optimality gaps of the 2-iteration and midway methods in small instances. Note that in Table 3, we also present the optimality gaps of the "NominalRepair" method. We propose the NominalRepair method to benchmark the performance of the 2-iteration method. We describe this method as follows. Initially, this method solves the nominal 2cPCKP, and if the solution obtained is not feasible with respect to the chance capacity constraint (10), then it tries to repair the selection of items chosen, by the nominal model, by removing items from this selection one by one until the remaining selection becomes feasible. At each iteration, this method removes an item with the least profit whose successors are not in the selection. This is for maintaining the integrity of precedence relations satisfied in the nominal solution.

Firstly, we observe in Table 3 that the optimality gaps of our approximate cutting plane methods are reasonably small in small instances; the highest average optimality gap that we find of our approximate methods

Table 2: The average number of items eliminated by pre-processing for different values of p_i, b, d in the large instances.

	b	d	$\|I\| = 1000$	$\|I\| = 2000$	$\|I\| = 4000$	$\|I\| = 8000$
	50	1	0	0	10	42
	50	5	398	901	1908	3924
	50	10	432	940	1944	4009
	100	1	0	0	0	3
$p_i = UC$	100	5	240	726	1743	3762
	100	10	318	825	1838	3866
	200	1	0	0	0	0
	200	5	0	371	1392	3406
	200	10	95	603	1620	3639
	50	1	0	0	3	14
	50	5	393	890	1887	3917
	50	10	431	939	1937	3969
	100	1	0	0	0	2
$p_i = WC$	100	5	243	737	1746	3745
	100	10	323	819	1853	3861
	200	1	0	0	0	0
	200	5	0	382	1386	3362
	200	10	58	604	1599	3599
	50	1	0	0	0	0
	50	5	390	881	1870	3880
	50	10	430	936	1935	3926
	100	1	0	0	0	0
$p_i = SC$	100	5	233	735	1725	3735
	100	10	313	810	1810	3841
	200	1	0	0	0	0
	200	5	0	388	1368	3369
	200	10	70	597	1596	3598
Ave. (#)			169	450	1122	2447
Ave. (%)			17%	23%	28%	31%

Table 3: The average optimality gaps by the 2-iteration, midway and NominalRepair methods on the small instances.

	d	p_i	NominalRepair	2-iteration	Midway
	1	UC	19.52%	0.34%	**0.00%**
	1	WC	25.20%	1.35%	**0.01%**
	1	SC	33.19%	0.72%	**0.00%**
	5	UC	4.54%	**0.00%**	**0.00%**
$\|I\|$=10	5	WC	10.80%	**0.16%**	**0.16%**
	5	SC	13.08%	0.38%	**0.21%**
	10	UC	4.54%	**0.00%**	**0.00%**
	10	WC	10.80%	**0.16%**	**0.16%**
	10	SC	13.08%	0.38%	**0.21%**
	1	UC	4.01%	0.05%	**0.00%**
	1	WC	4.35%	0.04%	**0.00%**
	1	SC	4.55%	0.06%	**0.00%**
	5	UC	2.32%	0.02%	**0.00%**
$\|I\|$=50	5	WC	2.68%	0.06%	**0.00%**
	5	SC	3.64%	0.01%	**0.00%**
	10	UC	2.32%	0.02%	**0.00%**
	10	WC	2.68%	0.06%	**0.00%**
	10	SC	3.64%	0.01%	**0.00%**
	1	UC	2.31%	**0.00%**	**0.00%**
	1	WC	2.31%	**0.00%**	**0.00%**
	1	SC	2.56%	0.01%	**0.00%**
	5	UC	1.06%	0.04%	**0.00%**
$\|I\|$=100	5	WC	1.37%	0.06%	**0.00%**
	5	SC	1.38%	**0.00%**	**0.00%**
	10	UC	1.06%	0.04%	**0.00%**
	10	WC	1.37%	0.06%	**0.00%**
	10	SC	1.38%	**0.00%**	**0.00%**

is less than 1.4%, which is realized by the 2-iteration method. Recall the discussion in Section 4.2.2, where we show that the midway method will be superior to the 2-iteration method in terms of the quality of the solutions it produces. Note that we can confirm this superiority in each of the instances presented in Table 3. When we look at the performance of the NominalRepair method, which can reach an optimality gap of 33%, we see that similar to the 2-iteration method, the NominalRepair method is sensitive to a small number of items. However, this sensitivity seems more prevalent for the NominalRepair method. This indicates the superiority of the 2-iteration method over the NominalRepair method.

In each of the instances with $|I| \geq 1000$, we find that neither the 2-iteration nor the midway method produces solutions that are of worse quality than the optimal solutions. In other words, our approximate cutting plane methods have no optimality gaps in the large instances. As we explain in introducing the instances in Section 5, we expect that as the number of items increases in instances, the approximation errors made by the Taylor approximations of the function $\sqrt{\sum_{i\in I} \sigma_i^2 y_i^2}$ would be less important and the solutions that our approximate methods find would be closer to optimal solutions. Considering the optimality gap performance of our approximate methods in small and large instances, we conclude that these methods can have very small optimality gaps, and when the number of items is large, they can be as effective as an exact approach in finding high quality solutions.

Let us now focus on the computational performance of our approximate methods. In this investigation, we use the performance of the exact cutting plane method and the original quadratic formulation solved directly by Gurobi as benchmarks. For small instances ($|I| \leq 100$ instances), we confirm the efficiency of the quadratic formulation; the average computation time on these instances is found to be 0.2 seconds. Our approximate

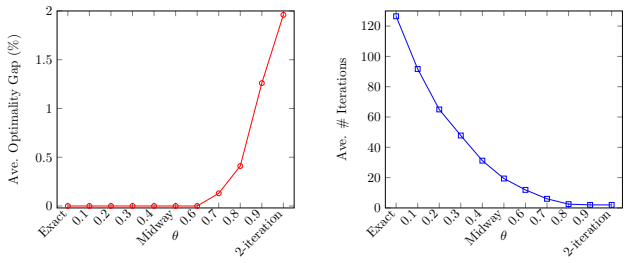
Table 4: The average computational performance of the cutting plane methods and the quadratic formulation

for different values of p_i and $|I|$ in the large instances, averaged over all settings for b and d.

		2-iteration		Midway		Exact (Linear)		Quadratic	
		Time (secs)	#Iter	Time (secs)	#Iter	Time (secs)	#Iter	Time (secs)	Opt
	$p_i = UC$	**53.7**	**2**	175.5	16.1	7744.9	186.8	4966.3	31%
$\|I\| = 1000$	$p_i = WC$	**3.5**	**2**	254.7	34.7	22 753.7	476.6	5533.4	25%
	$p_i = SC$	**3.7**	**2**	1068.8	53.5	21 587.1	432.6	5528.4	26%
	$p_i = UC$	**94.2**	**2**	1870.1	9.9	10 510.1	168.4	5039.0	33%
$\|I\| = 2000$	$p_i = WC$	**4.4**	**2**	166.1	19.5	17 034.8	469.1	5454.3	26%
	$p_i = SC$	**4.6**	**2**	829.5	17.9	22 801.0	547.5	5864.4	21%
	$p_i = UC$	**7.8**	**2**	36.6	5.6	5247.7	89.7	4103.9	45%
$\|I\| = 4000$	$p_i = WC$	**7.8**	**2**	70.3	6.7	4846.1	126.4	4959.8	33%
	$p_i = SC$	**7.5**	**2**	84.8	7.4	8317.8	133.0	4765.5	36%
	$p_i = UC$	**11.7**	**2**	44.4	4.4	214.5	17.1	3241.0	68%
$\|I\| = 8000$	$p_i = WC$	**12.5**	**2**	55.5	4.9	624.9	20.8	4528.4	40%
	$p_i = SC$	**18.1**	**2**	124.8	5.8	2202.9	28.8	4595.8	40%

cutting plane methods that use linear formulations are also able to solve the small instances very efficiently. However, given that these methods have optimality gaps, although very small (see Table 3), on small instances, we conclude that solving the quadratic formulation directly is more advantageous than our approximate cutting plane methods on these small instances.

Now, let us focus on the large instances ($|I| \geq 1000$ instances). In Table 4, we present the average computational results for the cutting plane methods and the quadratic formulation. First of all, we observe in the first two columns of Table 4 that the 2-iteration method takes only two iterations to find feasible solutions, and without requiring much computational time. The midway method, which uses the equal-mix of Taylor approximations of the 2-iteration and exact methods, seems to be able to incorporate the computational efficiency advantage of the 2-iteration method; we find out that the midway method requires slightly more computational time than the 2-iteration method, however, much less than the exact method. In Table 4, we see that the exact cutting plane method (under the column "Excact (Linear)") can be quite a computational burden. We know that the 2-iteration (also the midway) method has no optimality gaps in any of these instances. Since we allocate one hour per iteration, the total running time allotted to the 2-iteration method is two hours for every instance. In order to make a fair comparison, we allow two hours to the quadratic formulation. The last two columns of Table 4 give the performance of the quadratic formulation. Since this formulation is not solved to optimality always, under the column 'Opt' we present the percentage of the instances that are solved to optimality. When we compare the performance of our approximate methods to the quadratic formulation, we can clearly see the superiority of our approximate cutting plane methods. Therefore, we conclude that for large instances, methods such as the 2-iteration method which use linear formulations can be needed for overcoming the challenge of solving the nonlinear formulation (9)-(14). However, when we compare the performance of the exact cutting plane method to the quadratic formulation in Table 4, the advantage of the exact cutting plane method is not clear, considering the high computational requirements of this method and the low optimality gaps (with an overall average gap below 1%) found of the solutions that are not solved to optimality by the quadratic formulation. It must be noted that the exact cutting plane method does not provide a feasible solution before its final iteration; it can only provide an upper bound on the objective. Given that we cannot predict in how many iterations this method will finally deliver a feasible (and optimal) solution, in situations in which feasible



(a) Effect on the Optimality Gaps

(b) Effect on the Number of Iterations

Figure 1: The effect of θ in the performance of cutting plane methods using the first-order approximation of $\sqrt{\sum_i \sigma_i^2(\theta y_i + (1-\theta)y_i^2)}$ to produce approximate feasibility cuts ($|I| = 50, b = 5, d = 1, p_i = WC$, $\sigma_i \sim U[1, 800]$, $\forall i$).

solutions must be found under a given time limit, this method might be impractical. On the other hand, even if the quadratic formulation cannot be solved to optimality up to a given time limit, it would provide feasible solutions. In this case, when the solutions found by the quadratic formulation have considerably low optimality gaps, solving the quadratic formulation might be preferred over the exact cutting plane method.

We discuss in Section 4.2.2 that an infinite number of convex-concave mixture methods that use the first-order linear approximation of $\sqrt{\sum_i \sigma_i^2(\theta y_i + (1-\theta)y_i^2)}$, for $\theta \in [0, 1]$, to produce approximate feasibility cuts can be proposed. As θ approaches zero, the approximate method will get closer to the exact method. This means that solutions might improve in quality, however, this can be a trade-off from computational efficiency. On the other hand, as θ approaches one, the approximate method will get closer to the 2-iteration method, which is able to converge to feasible solutions within two iterations. Therefore, parameter θ can be used as a leverage to balance the solution quality and computational efficiency. Tables 3-4 reveal how the midway method which lets $\theta = 0.5$ balances the speed advantage of the 2-iteration method and the solution quality advantage of the exact method. In order to illustrate this leveraging function of θ also for different values of θ, we present Figure 1, where the effect of θ on the solution quality and computational efficiency of the resulting approximate cutting plane method is shown on a small instance.

In Figure 1(a), we can observe the effect of θ on the optimality gaps of the solutions obtained with the resulting methods, whereas in Figure 1(b), we can see how θ affects the number of iterations that the resulting methods take for finding feasible solutions. We recognize in Figure 1(b) that there is a computational advantage in increasing θ, which increases the similarity of the approximations to those of the 2-iteration method. However, in Figure 1(a) we can witness how this comes at a cost from the solution quality. This indicates that θ can be adjusted according to the desired level of solution quality and computational time available.

Based on the performance of our approximate methods in small and large instances, as described by Table 1, it seems that our approximate methods are able to find feasible solutions with very small optimality gaps quickly. However, as we point out in Section 4.2.2, the optimality gaps of our approximate methods can be potentially higher in instances where the variances of the uncertain item weights are very high, or when these variances are negatively correlated to the means of the uncertain item weights. In what follows, we explore the optimality gaps of our approximate methods in these special cases. In doing this, we use a small instance with

$|I| = 50$, $d = 1$ and $p_i = WC$ to better detect the effect on the optimality gaps.

In Table 1, we let $\sigma_i \sim U[1, 200]$, $\forall i$, which models the standard deviation of $\xi_i - w_i$ random variables. In Figure 2, we consider different distributions of higher variance for modeling the standard deviation of $\xi_i - w_i$ random variables, and show how the optimality gaps of the 2-iteration and midway methods can be affected by the increase in σ_i values. In all the distributions considered for modeling σ_is in Figure 2, we find that the midway method remains optimal, however, the 2-iteration method reaches an optimality gap of nearly 2%, when $\sigma_i \sim U[1, 800]$, $\forall i$.

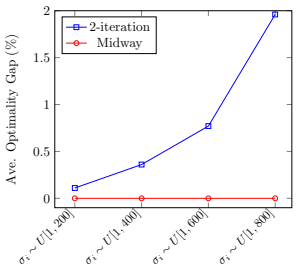


Figure 2: The effect of σ_i distribution in the optimality gap performance of the 2-iteration and midway methods ($|I| = 50$, $b = 5$, $d = 1$ and $p_i = WC$).

Next, we consider a very high variance and negatively correlated scenario for modeling σ_i, $\forall i$. In this scenario, we first sample μ_i per item i from $N(250, 200^2)$, as in Table 1, and then we let $\sigma_i = 0$, for any item i with $\mu_i \geq 250$, otherwise we let $\sigma_i = 800$. Note that in the instance setting with $\sigma_i \sim U[1, 800]$, $\forall i$ the 2-iteration method gives an average optimality gap of 1.96%, however, in this high-variance and negatively correlated scenario we find that the average optimality gap reaches to 12.72%. On the other hand, the midway method, which has no optimality gap when $\sigma_i \sim U[1, 800]$, $\forall i$, gives an average optimality gap of 7.68% in this special variance scenario. This confirms that if the variances of uncertain item weights are very high, and/or these variances are negatively correlated to the means of the uncertain item weights, our approximate methods, in particular the 2-iteration method, can perform relatively worse.

This high optimality gap of the 2-iteration method can be reduced with a post-processing procedure which takes the y solutions of the method as input and solves for the corresponding optimal x solutions. Note that solving the 2cPCKP for optimal x given y solutions is a linear program that can be solved efficiently, which we can consider it as a post-processing procedure. We find that this procedure improves the 2-iteration method solutions by 0.14% and 0.77% on the $\sigma_i \sim U[1, 600]$ and $\sigma_i \sim U[1, 800]$ instance settings reported in Figure 2, respectively. For the negatively correlated scenario considered above, we find that this post-processing improves the 2-iteration method solutions by 1.2%. This shows that the optimality gaps of the 2-iteration method can be attributed to the suboptimal x selections for the items which are not picked with the non-preferred method. To investigate if the optimality gaps of the 2-iteration method are related to the suboptimal y selection, we solve the quadratic formulation of the 2cPCKP where the x selection is fixed to the x solutions of the 2-iteration method. In this, we do not find that the solution quality of the 2-iteration method can be improved.

5.3 Performance of the Cover Cuts

In this section, we present Table 5 to show the sensitivity of the exact cutting plane method to the problem characteristics and to demonstrate the effect of adding cover cuts in improving the computational efficiency of the exact cutting plane method. For this, we use the instances with $|I| = 8000$, and focus on the running times and the number of iterations that the exact cutting plane method takes to converge to optimal solutions. In producing these results, we let the number of trials used in the constructive heuristic that finds cover cuts to be $|I|$, and we do not restrict the number of cuts that can be added. The first two columns of Table 5 give the computational performance of the exact cutting plane method implementation which does not use any cover cuts. On the other hand, the last two columns of Table 5 relate to the performance of the implementation which generates and adds cover cuts within the branch-and-bound tree. In Table 5, we can observe the effect of p_i, d and b parameters on the computational performance of the exact cutting plane method implementations. In Table 5, the results presented for different values of p_i correspond to the average performance measures over all settings for b and d, and the results presented for different values of d correspond to the average performance measures over all settings for b and p_i, and lastly the results presented for different values of b correspond to the average performance measures over all settings for p_i and d.

Table 5: The average computational performance of the exact cutting plane method and cover cuts in $|I| = 8000$ instances for different values of p_i, d and b. The results presented for different values of p_i (d, or b) correspond to the average performance measures over all settings for b and d (b and p_i, or d and p_i).

	Exact		Exact (+ Cover Cuts)	
	Time (secs)	#Iters	Time (secs)	#Iters
$p_i = UC$	**214.5**	**17.1**	575.1	18.4
$p_i = WC$	624.9	20.9	**161.2**	**12.9**
$p_i = SC$	2203.0	28.8	**656.7**	**17.0**
$d = 1$	689.2	18.4	**477.4**	**15.1**
$d = 5$	**92.3**	**10.6**	351.6	17.1
$d = 10$	2433.9	39.4	**651.5**	**16.7**
$b = 50$	266.5	20.8	**142.8**	**13.8**
$b = 100$	194.6	12.1	**182.6**	**14.2**
$b = 200$	2563.6	34.0	**1149.2**	**21.1**

Observing the overall average performance of the two implementations in all parameter settings of $|I| = 8000$ instances, we find that with the implementation without cover cuts the average running time is 1033.83 seconds, while the implementation with the cover cuts has an average running time of 489.14 seconds. This indicates a significant difference that cover cuts can make in improving the computational efficiency of the exact cutting plane method. However, in Table 5, we also note that cover cuts are not improving the computational efficiency for some specific instance settings, namely, in instances where $p_i = UC$ and $d = 5$. We observe that in these instances the running times of the exact cutting plane method are significantly shorter, compared to other settings. In other words, from the results in Table 5, we identify that cover cuts are useful in difficult instances in which the exact cutting plane takes much longer time to find optimal solutions. In instances where the precedence graphs are dense ($d = 10$), and low in depth ($b = 200$), and the item profits are strongly correlated to their weights ($p_i = SC$), the exact cutting plane method implementation without cover cuts requires a

Table 6: MIT undergraduate Mathematics courses.

Course No.	Course Name	Units	σ_i	Prerequisites
1	Calculus I	5-0-7	7	-
2	Differential Equations	5-0-7	5	-
3	Algebra I	3-0-9	15	-
4	Calculus II	5-0-7	5	Calculus I
5	Algebra II	3-0-9	24	Algebra I
6	Mathematics for Computer Science	5-0-7	24	Calculus I
7	Linear Algebra	4-0-8	6	Calculus II
8	Numerical Analysis	3-0-9	24	Calculus II, Differential Equations
9	Real Analysis	3-0-9	25	Calculus II
10	Introduction to Functional Analysis	3-0-9	19	Linear Algebra, Real Analysis
11	Probability and Random Variables	4-0-8	14	Calculus II
12	Introduction to Stochastic Processes	3-0-9	6	Probability and Random Variables
13	Fundamentals of Statistics	4-0-8	18	Probability and Random Variables
14	Commutative Algebra	3-0-9	20	Algebra II
15	Elliptic Curves	3-0-9	1	Algebra I
16	Lie Groups and Lie Algebras I	3-0-9	26	Algebra I, Real Analysis
17	Lie Groups and Lie Algebras II	3-0-9	20	Lie Groups and Lie Algebras I
18	Introduction to Topology	3-0-9	11	Real Analysis

running time of at least 30 minutes. We observe that in these difficult instances, cover cuts reduce the running times by more than 50%. As a consequence, we conclude that cover cuts can be deemed useful, especially in solving the challenging instances and in avoiding very long running times.

5.4 Modeling an Educational Setting with the 2cPCKP

Finally, we return to the personalized learning application that motivated the 2cPCKP and consider a student who is interested in studying courses from MIT's mathematics undergraduate major program. A complete list of courses offered in this program can be found at the catalog `http://student.mit.edu/catalog/m18a.html`. In this program, each course is associated with *units*. One unit is approximately equal to 14 study hours per term. The units for each course are displayed as a series of three numbers (e.g., 3-2-7). The numbers added together (e.g., 3+2+7) gives the total credit for the subject (e.g., 12). These three numbers represent units assigned for lectures and recitations, units assigned to laboratory, design, or field work, and units for outside preparation, respectively. In the course catalog, we can also see prerequisites of courses.

We assume that the student has 18 ($|I| = 18$) potential courses, given in Table 6, to select from and is interested in selecting some of these with the objective of maximizing the earned credits, while securing against being overloaded given the total number of study hours available. It must be noted that the reason we consider only a small set of courses is for illustration purposes such that we can show the setting and the results in a compact manner here. However, in practice, university students can indeed have a lot more options for courses. The student knows that the unit based study hour estimates of each course will be accurate for lectures, recitations and laboratory work. However, the student is aware that the estimates will not be accurate for outside preparation work. For example, the number of study hours required for preparation work on a course might depend on the student's personal learning pace.

We represent this situation with the 2cPCKP where we let $R = 1$. That is, for a course with units "3-2-7", we let $w_i = (3+2) \times 14$ and its profit to be $p_i = (w_i + \mu_i)/14$, while we consider that the total number of hours required for preparation work will be a normally distributed random variable with mean $\mu_i = 7 \times 14$. Note that in this example, item profits are strongly correlated to their weights. The standard deviations we see in

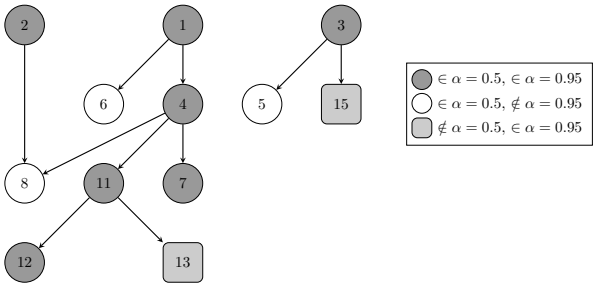


Figure 3: The nominal solution ($\alpha = 0.5$) versus the $\alpha = 0.95$ reliable solution.

Table 6 of the normally distributed variables per course are obtained by random sampling σ_i from $U[1, 14 \times 2]$. In practice, these uncertainty parameters can be obtained by using historical data, for example by using the student and lecturer evaluations from previous years. We set the total number of hours the student can spare to 1700 (i.e., $B = 1700$).

In this example, the optimal solution of the nominal model where $\alpha = 0.5$ is to select the courses numbered 1, 2, 3, 4, 5, 6, 7, 8, 11 and 12. Studying these courses will add up to a total of 120 credits. In this solution, the student is not concerned about the uncertainties involving the estimations of outside preparation work. If the student desires to be secured against not exceeding the study capacity with at least 75% reliability ($\alpha = 0.75$), then the optimal selection for the student will be 1, 2, 3, 4, 6, 7, 11, 12 and 15 summing to 108 credits. We see how this risk aware approach can lead the student to select less courses and earn fewer credits. Moreover, when we compare the courses selected in the nominal solution to the ones selected in the 75% reliability case, we see not only that some courses that are selected in the nominal solution are not selected in the 75% reliable solution (courses numbered 5 and 8), but there are also some courses that are selected in the 75% reliable solution (course numbered 15) but not so in the nominal solution. Now, let us consider that the student prefers 95% reliability. For this, the optimal selection will be 1, 2, 3, 4, 7, 11, 12, 13 and 15 with 108 credits in total. We observe that the dissimilarity of the 95% reliable optimal selection to the nominal solution is higher than that of the 75% reliable optimal selection.

We illustrate the nominal and 95% reliable solutions in Figure 3, which depicts the precedence relations among the selected courses. In Figure 3, courses represented with dark gray circles are the courses that are selected in both the nominal and $\alpha = 0.95$ reliable solutions, the ones represented with white circles are the courses that are selected in the nominal solution but not in the $\alpha = 0.95$ reliable solution, and the ones represented with light gray rectangles are the courses that are not selected in the nominal solution but in the $\alpha = 0.95$ reliable solution. We can grasp from this figure that when the uncertainty is incorporated, Course 15 with a very small variance involving the outside preparation estimate is considered as an alternative to Course 5 that is selected in the nominal solution. Note that both of these courses are successors of Course 3, which is selected in both the nominal and 95% reliable solutions. Also, it seems that Course 6 and Course 8 which have relatively high variances involving their outside preparation estimates are given up in the 95% reliable solution for Course 13 with a smaller variance. This educational-context example illustrates how the optimal decisions of a knapsack problem can differ depending on how the decision makers perceive risks under the presence of precedence relations among items.

5.5 Benchmarking the 2-iteration Method in Solving the Chance-constrained Binary Knapsack Problem with Normally Distributed Item Weights

In previous sections we have shown that the 2-iteration method is efficient for the 2cPCKP, in particular for solving large problem instances. In this section, we use the 2-iteration method for a special case of the 2cPCKP, with $R = 1$ and $E = \emptyset$, and we compare it with robust optimization–based methods from the literature. This special case can be interpreted as the chance-constrained binary knapsack problem in which all item weights are independently and normally distributed, which is studied in Han et al. (2016) and Joung & Lee (2020), and can be considered as the closest setting to our problem in the literature. We test our 2-iteration method on the same instances as Joung & Lee (2020) do, which are selected from the instances in Monaci et al. (2013). In order to make a fair comparison, we compare the performance of our 2-iteration method to the performance of the robust optimization–based methods reported in Joung & Lee (2020).

The test instances are defined by two parameters: the number of items $|I|$ within the set $\{100, 500, 1000, 5000\}$ and the instance type, which can be either strongly correlated (SC), or inversely correlated (IC), or subset sum (SS) type. For the details on how the item weights and profits are generated in each of these types, we refer the readers to Joung & Lee (2020). In Joung & Lee (2020), CPLEX is used to find optimal solutions to the quadratic formulation. However, the authors report that for the majority of the instances (in 99 out of 120 instances) optimal solutions are not found. Here, given the recent developments made by the MIP solvers, we use Gurobi 9.1 to solve the quadratic formulation, instead of using the optimal solutions reported in Joung & Lee (2020). In order to create a similar test environment as in Joung & Lee (2020), for each parameter setting we take 10 samples to generate instances, and used the time limit of 1800 seconds for Gurobi and our 2-iteration method. In our offline investigations, we realize that the chance of Gurobi to find optimal solutions within the given time limit increases a lot with a warm start approach. Using a simple construction heuristic which uses a priority measure of the items based on the item profits, and the uncertain item weight means and variances $(p_i/(w_i + \sqrt{\sigma_i}\Phi^{-1}(\alpha)))$, as we propose in Algorithm A2 in the Appendix for the 2cPCKP, we provide Gurobi the obtained lower bounds on the optimal values, and are able to find the optimal solutions to the most of the instances within the given time limit. In Joung & Lee (2020), the authors provide results for $\alpha = 0.9$ and $\alpha = 0.95$ cases. Our observations reveal that the case $\alpha = 0.90$ is a lot easier than the case $\alpha = 0.95$. For this reason, for the sake of brevity, focusing only on the $\alpha = 0.95$ case, we present our results in Table 7. The columns under 'J&L(2020)' and 'H(2016)' relate to the performance of the methods proposed in Joung & Lee (2020) and Han et al. (2016), respectively, and are directly taken from the performance investigation reported in Joung & Lee (2020). We note that the method proposed in Han et al. (2016) provides upper bounds, and it does not guarantee the feasibility of the solution. For this reason, only the computational times are reported for this method.

In Table 7, the 'Time' column under 'Gurobi' gives the average computation time per instance setting, including the ones that could not be solved to optimality. Below the column 'Obj/UB' under 'Gurobi', the first entry gives the average objective found of the instances which are solved to optimality, and the number inside parentheses gives the number of such instances out of 10 samples, while the second entry gives the average upper bounds for the instances that could not be solved to optimality. We observe that except for the large inversely correlated (IC) instances, the quadratic formulation is able to efficiently find optimal solutions. Only in 15 out of 120 instances generated, this formulation is not able to find optimal solutions within the time limit.

Table 7: The performance of the 2-iteration method, quadratic formulation, Joung & Lee (2020) and Han et al. (2016) for $\alpha = 0.95$. Time is measured in seconds, gap is measured in percentages.

		2-iteration		Gurobi		J&L(2020)		H (2016)
$\|I\|$	Type	Time	Gap	Time	Obj/UB	Time	Gap	Time
100	SC	0.08	0.00	0.18	3078.8(10)/-	0.2	0.01	4.8
	IC	0.16	0.00	1.24	2687.8(10)/-	0.1	0.00	4.0
	SS	0.01	0.07	0.17	2481.9(10)/-	0.1	0.04	3.9
500	SC	0.19	0.00	289.36	15781.3(9)/15730.0(1)	2.7	0.01	1534.4
	IC	3.39	0.04	1800.0	-/13379.4(10)	1.2	0.00	1511.7
	SS	0.01	0.00	0.02	12553.6(10)/-	0.8	0.02	1600.1
1000	SC	0.05	0.00	0.05	32011.6(10)/-	11.6	0.03	-
	IC	0.39	0.00	360.19	26980.4(8)/27222.5(2)	4.2	0.01	-
	SS	0.01	0.00	0.01	24939.7(10)/-	3.3	0.21	-
5000	SC	0.10	0.00	0.05	160452.4(10)/-	526.2	0.01	-
	IC	0.14	0.00	541.66	136087.6(8)/135739.6(2)	148.8	0.00	-
	SS	0.07	0.00	0.03	125734.8(10)/-	111.5	0.10	-

We remark that the warm start strategy that we use is an important factor in this performance. Moreover, the recent developments made by the MIP solvers can also be contributing to this improved performance.

We can clearly see in Table 7 that the 2-iteration method is able to produce optimal or near optimal solutions in a very short time in every instance. Its average gap over all instances is 0.01%. Moreover, as we note also in our earlier investigations in Section 5.2, we identify that the optimality gap of this method disappears as the instance size increases; we observe that the 2-iteration method has no optimality gaps for $|I| = 1000$ and $|I| = 5000$ instances. Compared to the robust optimization–based methods proposed in Joung & Lee (2020) and Han et al. (2016), we note that our method does not lose its computational efficiency as the instances become larger. Both of these methods have to solve more and more knapsack problems as the number of items in the instances increases, however, our method always requires two iterations of nominal knapsack problems, irrespective of the size of the instances. When we compare the performance of the 2-iteration method to the robust optimization–based method proposed in Joung & Lee (2020) for $|I| \geq 1000$ instances in Table 7, we can clearly see the advantage of our approximate cutting plane method; the 2-iteration method is significantly faster and finds solutions with no optimality gaps. This demonstrates the competitiveness of our approximate cutting plane method in solving the chance-constrained binary knapsack problem in which item weights are independently and normally distributed, especially for large instances.

6 Conclusions

In this paper, we introduce the two-choice precedence constrained knapsack problem (2cPCKP) in which two methods are available to place an item in the knapsack where one of the methods subject item weights to uncertainties. Our study is motivated by the subject (or subject unit) selection problem of students in personalized learning environments where students can learn either under teacher supervision or in a self-study mode by using online learning tools. We use a chance-constrained programming framework for the 2cPCKP and assume that uncertain item weights are independently and normally distributed. Under this assumption, we focus on a deterministic reformulation which has a capacity constraint with a nonlinear and convex function of the decision variables. By exploiting the convexity of this function, we propose an exact cutting plane method which iteratively converges to the optimal solution by refining the feasibility space with cuts that are

based on the first-order linear Taylor approximations of the function. The disadvantage of this method is that it can take many iterations until it converges. To supplement the exact cutting plane method, in this paper, we propose novel approximate cutting plane methods that converge quickly to feasible solutions. One of the approximate methods is able to find a feasible solution within two iterations for any given problem instance. In our extensive computational experiments with a set of small and large instances, we find that our approximate cutting plane methods have very small optimality gaps overall, and in large instances these methods become as effective as an exact approach in finding high quality solutions. Moreover, we demonstrate the efficiency of our approximate cutting plane methods in solving the chance-constrained binary knapsack problem with independent and normally distributed weights, by benchmarking against two robust optimization–based methods proposed in the literature. As in the 2cPCKP, we find that our methods are able to find optimal solutions to the large instances with at least 1000 items of this knapsack problem with no precedence constraints. In this paper, we also present new pre-processing procedures to eliminate items beforehand and cover cuts to improve the efficiency of our cutting plane methods. Our computational experiments demonstrate that the proposed pre-processing procedures become highly effective when the precedence relations among items are dense and cover cuts are useful for avoiding very long running times with the exact cutting plane method. We also provide a real-life based educational-context example for the 2cPCKP, where we illustrate the effect of incorporating risks in decision making under the precedence relations.

Endnotes

[1]The limitation of not considering the correlations between the study times of different subjects depends on how the data is used to fit parameters (mean and variance) to the distributions. For example, when the parameters of the distribution for a specific student studying a specific subject is fitted using a series of past performance of the same student in completing a similar type of activity, the limitations of this assumption will be very low. On the other hand, if we were to use data that comes from a large student population, then this assumption would be significantly limiting. In our paper, we assume that student specific data is available and therefore consider a situation in which the independence assumption is not limiting.

[2] It is worth mentioning that even in the situations where study times are not suitable to be modeled with normal distributions, the limitation of this assumption for our knapsack problem will not be significant, as long as the number of items to be selected is large enough, due to the central limit theorem. This is because in our formulation the feasibility of the problem concerns the sum of the total number of items selected (the capacity constraint), not the items individually.

References

Araujo, J. A. S., Santos, H. G., Gendron, B., Jena, S. D., Brito, S. S., & Souza, D. S. (2020). Strong bounds for resource constrained project scheduling: Preprocessing and cutting planes. *Computers & Operations Research*, *113*, 104782.

Aslan, A., Bakir, I., & Vis, I. F. A. (2020). A dynamic Thompson sampling hyper-heuristic framework for learning activity planning in personalized learning. *European Journal of Operational Research*, *286*(2), 673-688.

Bloom, B. S. (1968). Learning for mastery. *UCLA - CSEIP - Evaluation Comment*, *1*(2), 1-5.

Boland, N., Bley, A., Fricke, C., Froyland, G., & Sotirov, R. (2012). Clique-based facets for the precedence constrained knapsack problem. *Mathematical Programming*, *133*, 481-511.

Boyd, E. A. (1993). Polyhedral results for the precedence-constrained knapsack problem. *Discrete Applied Mathematics*, *41*, 185-201.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Calafiore, G. C., & Ghaoui, L. E. (2006). On distributionally robust chance-constrained linear programs. *Journal of Optimization Theory and Applications*, *130*, 1-22.

Caserta, M., & Vos, S. (2019). The robust multiple-choice multidimensional knapsack problem. *Omega*, *86*, 16-27.

Charnes, A., & Cooper, W. W. (1959). Chance-constrained programming. *Management Science*, *6*, 1-140.

Chicoisne, R., Espinoza, D., Goycoolea, M., Moreno, E., & Rubio, E. (2012). A new algorithm for the open-pit mine production scheduling problem. *Operations Research*, *60*, 517-528.

Duran, M. A., & Grossmann, I. E. (1986). An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, *36*, 307-339.

Eiken, O. (2011). The kunskapsskolan ("the knowledge school"): A personalised approach to education. *CELE Exchange, Centre for Effective Learning Environments, OECD Publishing, Paris*, *1*, 6.

European Commission. (2019). *10 trends transforming education as we know it.* `https://op.europa.eu/s/ofot`.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of np-completeness*. San Francisco: W. H. Freeman and Company.

Haase, K., Latteier, J., & Schirmer, A. (1999). Course planning at lufthansa technical training: Constructing more profitable schedules. *Interfaces*, *29*, 95-109.

Han, J., Lee, K., Lee, C., Choi, K.-S., & Park, S. (2016). Robust optimization approach for a chance-constrained binary knapsack problem. *Mathematical Programming*, *157*, 277-296.

Hijazi, H., Bonami, P., & Ouorou, A. (2013). An outer-inner approximation for separable mixed-integer nonlinear programs. *INFORMS Journal on Computing*, *26*(1), 31-44.

Johnes, J. (2015). Operational research in education. *European Journal of Operational Research*, *243*(3), 683-696.

Johnson, D. S., & Niemi, K. A. (1983). On knapsacks, partitions, and a new dynamic programming technique for trees. *Mathematics of Operations Research*, *8*, 1-14.

Joung, S., & Lee, K. (2020). Robust optimization-based heuristic algorithm for the chance-constrained knapsack problem using submodularity. *Optimization Letters*, *14*, 101-113.

Kannan, A., van den Berg, G., & Kuo, A. (2012). iSchedule to personalized learning. *Interfaces*, *42*(5), 437-448.

Kasapidis, G. A., Paraskevopoulos, D. C., Repoussis, P. P., & Tarantilis, C. D. (2021). Flexible job shop scheduling problems with arbitrary precedence graphs. *Production and Operations Management*, *30*(11), 4044-4068.

Kim, R., Olfman, L., Ryan, T., & Eryilmaz, E. (2014). Leveraging a personalized system to improve self-directed learning in online educational environments. *Computers & Education*, *70*, 150-160.

Kosuch, S., & Lisser, A. (2010). Upper bounds for the 0-1 stochastic knapsack problem and a b&b algorithm. *Annals of Operations Research*, *176*, 77-93.

Kristiansen, S., Sørensen, M., & Stidsen, T. R. (2011). Elective course planning. *European Journal of Operational Research*, *215*(3), 713-720.

Kunskapsskolen. (2021). *The KED program.* Retrieved from `http://kunskapsskolan.com/thekedprogram.4.1d96c045153756b0c14d5798.html`

Lagos, T., Armstrong, M., Homem-de Mello, T., Lagos, G., & Saure, D. (2020). A framework for adaptive openpit mining planning under geological uncertainty. *Optimization and Engineering*.

Lightfoot, J. M. (2006). Modular curriculum design using personal learning plans and reusable learning components. *Communications of the IIMA*, *6*, Article 6.

Lin, C. K. Y. (2009). Stochastic single-source capacitated facility location model with service level requirements. *International Journal of Production Economics*, *117*, 439-451.

Lu, M., Nakao, H., Shen, S., & Zhao, L. (2021). Non-profit resource allocation and service scheduling with cross-subsidization and uncertain resource consumptions. *Omega*, *99*, 102191.

Lyu, G., Chou, M. C., Teo, C.-P., Zheng, Z., & Zhong, Y. (2022). Stochastic knapsack revisited: The service level perspective. *Operations Research*, *70*(2), 729-747.

Monaci, M., & Pferschy, U. (2013). On distributionally robust chance-constrained linear programs. *SIAM Journal on Optimization*, *23*(4), 1956–1982.

Monaci, M., Pferschy, U., & Serafini, P. (2013). Exact solution of the robust knapsack problem. *Computers & Operations Research*, *40*(11), 2625-2631.

Morton, D. P., & Wood, R. K. (1998). On a stochastic knapsack problem and generalizations. *In: Woodruff D.L. (eds) Advances in Computational and Stochastic Optimization, Logic Programming, and Heuristic Search. Operations Research/Computer Science Interfaces Series, Springer, Boston, MA.*, *9*, 149-168.

Nansheng, P., & Qichen, M. (2022). Resource allocation in robust scheduling. *Journal of the Operational Research Society*, *0*(0), 1-18.

NCES. (2020). *Education expenditures by country.* Retrieved from `https://nces.ed.gov/programs/coe/indicator_cmd.asp`

Nossack, J. (2022). Therapy scheduling and therapy planning at hospitals. *Omega*, *109*, 102594.

Ohanian, S. (1999). *One size fits few: The folly of educational standards.* 361 Hanover Street, Portsmouth, NH 03801-3912: Heinemann.

Pane, J. F., Steiner, E. D., Baird, M. D., Hamilton, L. S., & Pane, J. D. (2017). *Informing progress: Insights on personalized learning implementation and effects.* Retrieved from `https://www.rand.org/content/dam/rand/pubs/research_reports/RR2000/RR2042/RAND_RR2042.pdf`

Pillay, N. (2014). A survey of school timetabling research. *Annals of Operations Research*, *218*, 261-293.

Prain, V., Cox, P., Deed, C., Dorman, J., Edwards, D., Farrelly, C., ... Yager, Z. (2013). Personalised learning: Lessons to be learnt. *British Educational Research Journal*, *39*(4), 654-676.

Prekopa, A. (2003). Probabilistic programming. *Handbooks in Operations Research and Management Science*, *10*, 267-351.

Qiu, R., Sun, Y., & Sun, M. (2022). A robust optimization approach for multi-product inventory management in a dual-channel warehouse under demand uncertainties. *Omega*, *109*, 102591.

Samavati, M., Essam, D., Nehring, M., & Sarker, R. (2018). A new methodology for the open-pit mine production scheduling problem. *Omega*, *81*, 169-182.

Samphaiboon, N., & Yamada, T. (2000). Heuristic and exact algorithms for the precedence-constrained knapsack problem. *Journal of Optimization Theory and Applications*, *105*, 659-676.

Shaw, D. X., & Cho, G. (1998). The criticalitem, upper bounds, and a branchandbound algorithm for the tree knapsack problem. *Networks*, *31*, 205-216.

Smilowitz, K., & Keppler, S. (2020). On the use of operations research and management in public education systems. *INFORMS TutORials in Operations Research*, *1*(4), 84-105.

Tonissen, D. D., van den Akker, J. M., & Hooggeveen, J. A. (2017). Column generation strategies and decomposition approaches for the two-stage stochastic multiple knapsack problem. *Computers & Operations Research*, *83*, 125-139.

UNESCO. (2017). *Training tools for curriculum development: Personalized learning.* Retrieved from `https://unesdoc.unesco.org/ark:/48223/pf0000250057.locale=en`

Van de Leensel, R. L. M. J., van Hoesel, C. P. M., & van de Klundert, J. J. (1999). Lifting valid inequalities for the precedence constrained knapsack problem. *Mathematical Programming*, *86*, 161-185.

Weintraub, A., & Vera, J. (1991). A cutting plane approach for chance constrained linear programs. *Operations Research*, *39*, 776-785.

Yoo, J., & Garcia-Diaz, A. (2008). Cost-effective selection and multi-period scheduling of pavement maintenance and rehabilitation strategies. *Engineering Optimization*, *40*, 205-222.

You, B., & Yamada, T. (2007). A pegging approach to the precedence-constrained knapsack problem. *European Journal of Operational Research*, *183*, 618-632.

Appendices

A Heuristics for the 2cPCKP

We propose two heuristics for finding feasible solutions to the 2cPCKP.

Topological Sorting Heuristic

Our first heuristic is based on topological sorting of items. Topological orderings are frequently used in producing feasible solutions quickly to large problems that involve many precedence constraints (see Chicoisne et al. (2012) and Samavati et al. (2018)). The advantage of this simple heuristic is that it is able to produce a feasible solution very efficiently.

A topological order of a directed and acyclic graph (DAG), is an ordering of vertices in which vertex i must appear before vertex j in the order, if there is a directed edge from vertex i to vertex j. In the precedence graph $G = (I, E)$, there can be more than one topological ordering. Let π be any topological order of items I, found by a topological sorting procedure in the precedence graph $G = (I, E)$. Let $\pi(k)$ denote the item in the kth place of π. As π is a topological order, for any k, the selection of items $\{\pi(1), \pi(2), ..., \pi(k)\}$ will be feasible with

respect to the precedence constraints (see constraints (12)). The proposed heuristic is a constructive heuristic which initially starts from an empty solution $\mathcal{S} = \emptyset$, for the selection of items to be placed in the knapsack, and iteratively places new items to the selection $\mathcal{S}$ from the topological order, for letting $x_i = 1, \forall i \in \mathcal{S}$, until placing the considered item will not be feasible. Specifically, at iteration k, the selection $\mathcal{S}$ will be $\{\pi(1), \pi(2), ..., \pi(k)\}$. At any iteration of the heuristic, the selection $\mathcal{S}$ will be feasible if it does not violate the constraints (10), (11) and (13), as by construction it does not violate the constraints (12). Bear in mind that the selection $\mathcal{S}$ does not determine the decisions on the selection methods, however, feasibility also depends on the decision variables $y_i, \forall i \in \mathcal{S}$ which involve uncertain item weights. Note that different from the topological sorting heuristics proposed for PCKP, here we incorporate the uncertain item weights. In Algorithm A1, we describe the procedure we use for checking the feasibility, which also determines for which of the items the non-preferred method will be used.

Given a selection $\mathcal{S}$ for which we let $x_i = 1, \forall i \in \mathcal{S}$, we know that we should use the non-preferred method in $n = \lceil R|\mathcal{S}| \rceil$ of the items in $\mathcal{S}$. As we assume that $\mu_i > 0$ and $\sigma_i > 0$ for any item i, considering items for subjecting them to uncertainty more than necessary will be always suboptimal. However, in optimally determining exactly for which of the n items in $\mathcal{S}$, we should use the non-preferred method, we need to consider both the means and variances of the uncertain weights of the items in $\mathcal{S}$. For the nominal 2cPCKP, where $\alpha = 0.5$, we can optimally let the first n items with smallest means to be selected with the non-preferred method. We must note that for the case that $\alpha > 0.5$, this procedure may not be the optimal strategy to determine which items should be selected with the non-preferred method, for example when items with smaller uncertain weight means have high variances. To efficiently decide if a given selection $\mathcal{S}$ is feasible with respect to the constraints (10), (11) and (13), in Algorithm A1 we use a modified version of this simple strategy, by integrating the effects by the variances into the uncertain item weight means. We must note that the procedure described in Algorithm A1 is stricter than necessary; this procedure can identify an item selection that does not violate the constraints (10), (11) and (13) as infeasible, because of selecting a less advantageous set of items for the non-preferred method, however, it will never identify a selection that violates the constraints (10), (11) and (13) as a feasible selection. For this reason, when our heuristic finds a feasible selection $\mathcal{S}$, its profit would be a lower bound for the 2cPCKP.

Note that if at a given iteration of this heuristic, the selection $\mathcal{S} \cup \{\pi(k)\}$ is found to be infeasible, by the described approximate procedure, then any selection $\mathcal{S}'$ which contains $\mathcal{S} \cup \{\pi(k)\}$ will be also found infeasible by the procedure. Therefore, the heuristic can terminate when its attempt to add a new item to the selection fails.

Algorithm A1 Pseudocode of the feasibility checking procedure

Given a selection of items $\mathcal{S}$, let $k = |\mathcal{S}|$, $n = \lceil Rk \rceil$ and $\bar{\sigma^2} = \sum_{i \in I} \sigma_i^2 / |I|$;
Sort the items in $\mathcal{S}$ such that $\mu_{\mathcal{S}(1)} + \Phi^{-1}(\alpha)\sqrt{\sigma^2_{\mathcal{S}(1)} + (n-1)\bar{\sigma^2}} \leq \mu_{\mathcal{S}(2)} + \Phi^{-1}(\alpha)\sqrt{\sigma^2_{\mathcal{S}(2)} + (n-1)\bar{\sigma^2}} \leq ... \leq \mu_{\mathcal{S}(k)} + \Phi^{-1}(\alpha)\sqrt{\sigma^2_{\mathcal{S}(k)} + (n-1)\bar{\sigma^2}}$;
if $\sum_{l \in \mathcal{S}} w_l + \sum_{l=1}^{n} \mu_{\mathcal{S}(l)} + \Phi^{-1}(\alpha)\sqrt{\sum_{l=1}^{n} \sigma^2_{\mathcal{S}(l)}} \leq B$ **then**
    Return feasible;
else
    Return infeasible;
end if

A Randomized Construction Heuristic: Using Aggregated Profit to Weight Ratios in the Precedence Graph

Additionally, we develop a randomized construction heuristic by getting inspiration from the approach in

Shaw & Cho (1998) proposed for the tree knapsack problem, which assumes the tree structure on the precedence graph such that every item i can have at most one immediate predecessor. We expand this to general precedence graphs and extend its use to the 2cPCKP by incorporating uncertain item weights. In the heuristic, a feasible solution is iteratively constructed by using the potential measures of items that are based on the aggregated profit to weight ratios defined by the successors of the items. The motivation of this heuristic is that picking item i gives the potential to reach its successors $S(i) = \{j|(i,j) \in \mathcal{E}\}$. The calculation of item potential measures is described in Algorithm A2. In this calculation, given that $R \in [0,1]$ proportion of the selected items will be subject to uncertainty in the 2cPCKP, in calculating the aggregated weight of item i, we incorporate the uncertain item weights of the successors of the item, as if we would be letting $y_j = R$ for each $j \in S(i)$ (by relaxing the binary variable y_j to be continuous), when we let $x_i = 1$. So, μ_j of item $j \in S(i)$ makes a contribution of $R\mu_j$ in the aggregated weight of item i. It is less straightforward to measure the contribution of the variances $\sigma_j^2, \forall j \in S(i)$, as this depends on the entire selection (see constraint (11)). However, by considering that the contribution of $\sigma_j^2, \forall j \in S(i)$ in the aggregated weight of item i will be smaller than $\Phi^{-1}(\alpha)\sqrt{\sum_{j\in S(i)} R^2\sigma_j^2}$ in a solution which lets $x_i = 1$ and $y_j = R, \forall j \in S(i)$, and together recognizing the inequality that $\sum_{j\in S(i)} R\sqrt{\sigma_j} \leq \sqrt{\sum_{j\in S(i)} R^2\sigma_j^2}$, we consider that each item $j \in S(i)$ makes a contribution of $R\sqrt{\sigma_j}$ in the aggregated weight of item i.

Algorithm A2 Pseudocode for calculating the potentials

Initialize item potentials $\Pi(i) = 0, \forall i$;
Per item i, find $S(i)$ from $\mathcal{E}$ and let $W_i \leftarrow 0, P_i \leftarrow 0$;
for $i \in I$ **do**
  for $j \in S(i)$ **do**
    $W_i \leftarrow W_i + w_j + R(\mu_j + \sqrt{\sigma_j}\Phi^{-1}(\alpha))$, $P_i \leftarrow P_i + p_j$;
  end for
  $\Pi(i) \leftarrow P_i/W_i$;
end for
return Π

We describe how this heuristic works as follows. Initially, this heuristic starts with an empty selection. In the construction process, at each iteration, a new item is tried to be added to the selection through three types of moves: i) random-depth, ii) greedy-depth and iii) greedy-general moves. At each iteration, one of these moves is selected based on pre-assigned probabilities. The random-depth move tries to add an item that is an immediate successor of one of the items that are already in the selection. In selecting an item that is already in the selection, for the consideration of its immediate successors, a random approach is followed, and in searching among the successors, the heuristic adds the first item that is found feasible. Differently from the random-depth move, the greedy-depth and greedy-general moves use the potential measures of items when considering an item to add to the selection. The greedy-depth move tries to add the item with the highest potential which is an immediate successor of one of the items that are already in the selection, whereas the greedy-general move does not require that the added item is an immediate successor of one of the items that are in the current selection. Note that the greedy general move makes it possible to add items that have no predecessors and in the first iteration this move has to be used always. Note also that in this heuristic, the selection is kept feasible throughout the construction process. For checking if a given selection of items is feasible or not, the approximate procedure described in Algorithm A1 is used. The construction process ends when the number of iterations limit, or the limit on the number of iterations in which no new items are added to the selection is reached. As the selection maintained at each iteration is feasible, the final selection also provides a feasible

Table A1: The performance of two proposed heuristics averaged over all settings for b, d and p_i.

	Topo. Sorting		Rand. Construction	
$\|I\|$	Run Time (secs)	Ave. Gap (%)	Run Time (secs)	Ave. Gap (%)
1000	0.03	9.93%	7.70	6.60%
2000	0.13	7.04%	31.33	5.35%
4000	0.59	5.20%	248.94	4.19%
8000	2.73	4.37%	2308.11	3.53%

Table A2: The average optimality gap (%) of the randomized construction heuristic in $|I| = 1000$ instances.

b	d	$p_i = UC$	$p_i = WC$	$p_i = SC$
50	1	23.17%	5.57%	5.80%
50	5	4.70%	1.33%	1.22%
50	10	2.92%	1.09%	1.08%
100	1	23.53%	6.13%	6.46%
100	5	7.23%	2.46%	2.00%
100	10	6.08%	1.89%	1.99%
200	1	20.71%	6.09%	6.33%
200	5	14.75%	4.13%	4.07%
200	10	10.95%	3.32%	3.07%

solution to the 2cPCKP.

B Performance of the Heuristics

We present in Table A1 the overall average performance of the topological sorting and randomized construction heuristics proposed for the 2cPCKP in large instances. It can be observed that the topological sorting heuristic is efficient in generating a feasible solution quickly, while the randomized construction heuristic can produce better quality solutions. However, as the randomized construction heuristic iteratively seeks to add new items to its solution one by one, its computational burden seems to increase with the increased number of items available ($|I|$). Therefore, for situations in which the number of items available is too large the topological sorting heuristic might be preferred to the randomized construction heuristic.

Table A1 shows that as the number of items increases in instances the suboptimality of the solutions produced by the heuristics decreases. In order to show the effect of other problem parameters in the extent of the suboptimality of the heuristic solutions, we present Table A2, which gives the optimality gaps of the randomized construction heuristic for $|I| = 1000$ instances in which the heuristic gives higher gaps (see Table A1). In Table A2, we see that in instances where the precedence graph is dense and item profits are correlated to their weights, the optimality gap of the randomized construction heuristic becomes small, as small as nearly 1.00%. However, when the item profits are not correlated at all to their weights and the density of precedence relations is low ($d = 1$), the suboptimality can reach above 20%. This demonstrates the sensitivity of the heuristic to problem characteristics.