A decomposition strategy for decision problems with endogenous uncertainty using mixed-integer programming

Olli Herrala[a], Tommi Ekholm[b], Fabricio Oliveira[a,*]

[a]*Department of Mathematics and Systems Analysis, Aalto University, School of Science, FI-00076 Aalto, Finland*
[b]*Finnish Meteorological Institute, Helsinki, Finland*

Abstract

Decision problems under endogenous uncertainty are still challenging to solve, despite the advances in solution methods and increasing computational power. A novel framework called Decision Programming solves such decision problems using off-the-shelf solvers by using influence diagrams to represent decision problems with decision-dependent probabilities, and then converting the influence diagram representation of the problem to a mixed-integer linear programming problem.

This paper extends Decision Programming to accommodate the other fundamental type of endogenous uncertainty: conditionally observed information. Multi-stage stochastic programming (MSSP) models use conditional non-anticipativity constraints (C-NACs) to represent such uncertainties, and we show how such constraints can be incorporated into Decision Programming models. This allows us to consider both types of endogenous uncertainty simultaneously. Additionally, we present a decomposition approach that enables including continuous decision variables, whereas the original formulation was restricted to discrete variables only; and also provides significant computational savings.

The extended framework is illustrated with two example problems. The first considers an illustrative multiperiod game and the second is a large-scale cost-benefit problem regarding climate change mitigation. Neither of these example problems could be solved with existing frameworks.

Keywords: stochastic programming, endogenous uncertainty, climate change mitigation

1. Introduction

Stochastic programming (SP) is one of the most widespread mathematical programming-based frameworks for decision-making under uncertainty. In general, SP casts decision problems subject to parametric uncertainty as deterministic equivalents in the form of large-scale linear or mixed-integer programming (LP/MIP) problems that can be solved with standard optimization techniques.

One important feature related to SP models is the scenario tree. A scenario tree is a structural description of the dynamics between the decision process and the random variables that model the uncertain elements. In this setting, the random variables are typically assumed to have discrete finite support, for which each possible realization and respective probability are known.

The literature has seen a wealth of applications that rely on SP and associated developments (Ruszczyński and Shapiro, 2003; Wallace and Ziemba, 2005). The most widespread form of SP problems is two-stage stochastic programming (2SSP), emerging from problems with only two stages in the scenario tree.

*Corresponding author

Settings with more than two stages are referred to as multi-stage problems. Multi-stage stochastic programming (MSSP) has likewise seen intensive developments in the last few decades, mostly motivated by developments in closely related fields such as optimal control and reinforcement learning. MSSP models arise at the intersection between dynamic programming and SP. Dynamic programming is a framework specialized for problems in which dynamics play a central role (Bellman, 1966), and its extension to problems involving stochastic processes is known as stochastic dynamic programming (SDP). MSSP problems are essentially SDP problems and can therefore utilize the sophisticated solution techniques available for SDP problems, such as those presented in Pereira and Pinto (1991) and Powell (2007). A significant limitation in dynamic programming, however, is the no-forgetting assumption stating that at each decision stage, the full history of the stochastic decision process must be remembered. Additionally, considering risk measures such as value-at-risk is challenging for approaches based on dynamic programming because the optimal strategy in such problems depends on not only the history of the process but also the decisions that would be made if the history was different.

A common assumption in SP models is that the stochastic processes, particularly the state probabilities and/or observed values, are not influenced by the previously made decisions. The uncertainty is thus *exogenous*. This is methodologically convenient, for the deterministic equivalent model has the same nature as its stochastic counterpart, retaining important characteristics such as linearity, or more generally, convexity. Likewise, in the context of dynamic (MSSP) problems, exogenously uncertain decision problems can be modeled as Markov decision processes, for which the Bellman optimality conditions hold.

In this paper, we focus on the much less explored class of stochastic problems presenting *endogenous* uncertainty. In this more general setting, the decisions made at previous stages can affect the uncertainty faced in later stages. This endogenous nature is present in several important problems that have been addressed in the literature, such as disaster resilience in road networks (Peeta et al., 2010) and oilfield planning (Apap and Grossmann, 2017). Nonetheless, the literature considering endogenously uncertain SP problems is still scarce, particularly in comparison to the body of SP literature, likely due to the much more challenging nature of these problems.

It is common to classify SP problems according to the nature of the endogenous structure arising in the decision problem. Hellemo et al. (2018) has proposed a taxonomy of such problems, classifying the endogenous uncertainties into two distinct types. In Type 1 problems, earlier decisions influence the later events' probability distribution (i.e., realizations and/or the probabilities associated with each realization). In Type 2 problems, the *information structure* is influenced by the decision-making. Specifically, the decisions influence when information about the stochastic process becomes available without influencing the probabilities or observable outcomes associated with the random variables. It is noteworthy that Type 2 is more common in the literature on SP under endogenous uncertainty. The authors also introduce Type 3, which combines the Type 1 and Type 2 endogenous uncertainties.

Our main contribution is to provide a framework general enough to address Type 3 endogenously uncertain SP (T3ESP) problems, while still retaining prior computationally favorable properties of the mathematical model such as linearity or convexity. Our proposed framework builds upon Decision Programming, a recent contribution by Salo et al. (2022), and expands its capabilities to consider Type 2 endogenous uncertainty. Decision Programming was designed to incorporate Type 1 endogenous uncertainties in the decision process in an intuitive way, even in problems where the no-forgetting assumption does not hold. Salo et al. (2022) presents a simple problem with Type 2 endogenous uncertainty but does not discuss this class of endogenous uncertainty in detail. This paper explores Type 2 endogenous uncertainty further, showing explicitly how Decision Programming can be enhanced to become a strong framework for T3ESP problems. Additionally, we

present a decomposition approach that not only allows for modeling continuous decision variables, but also provides significant computational savings. The formulation presented in Salo et al. (2022) is designed for discrete decisions, and the authors mention continuous decisions as a possible future extension.

This paper is structured as follows. In Section 2, we present an overview of multi-stage stochastic programming. In Section 3, Decision Programming is described in detail, starting from the formulation in Salo et al. (2022) and continuing with our extensions. In Section 4 we consider a larger-scale climate change cost-benefit analysis, and Section 5 concludes and provides directions for further development.

2. Modeling problems with endogenous and exogenous uncertainties

Problems with exogenous uncertainty have been researched since the introduction of stochastic programming by Dantzig (1955). Since then, research on stochastic programming has introduced recourse problems and the notion of multi-stage stochastic programming (MSSP). Walkup and Wets (1967) first introduce the term *stochastic program with recourse*, denoting problems where decisions and random elements alternate sequentially. After making the first-stage decision, a random event occurs and the second-stage recourse action is chosen accordingly. Ruszczyński and Shapiro (2003) and Birge and Louveaux (2011) present more recent overviews of the stochastic programming methodology.

Solution approaches for MSSP are often based on formulating the deterministic equivalent problem using a scenario tree, as described in, e.g., Ruszczyński (1997). A scenario tree represents the structure of the uncertain decision process, and non-anticipativity constraints (NACs) are employed to enforce the information structure in the formulation. NACs state that a decision must be the same for two scenarios if those scenarios are indistinguishable when making the decision.

Notable applications of MSSP include financial portfolio management (Yin and Han, 2020) and energy systems planning (Lu et al., 2019). Additionally, supply chain (Alonso-Ayuso et al., 2003) and production (Goel and Grossmann, 2006) planning have an inherent structure that can be formulated as an MSSP problem, with multiple stages connected through, e.g., inventory constraints, resulting in large-scale problems. To alleviate the computational challenges, numerous heuristics and approximation methods have been proposed. These include knapsack-based methods (Huang and Ahmed, 2009) and (Lagrangian) dual decomposition (Carøe and Schultz, 1999).

In endogenously uncertain problems, the decisions can affect the timing, event probabilities, or outcomes of uncertain events further in the process. As previously discussed, endogenous uncertainty is often divided into *decision-dependent probabilities* (Type 1) and *decision-dependent information structure* (Type 2) (Hellemo et al., 2018). In this context, information structure often refers to when the realization of each uncertain event is observed, if ever. In contrast, exogenously uncertain problems have a fixed information structure with the timing of observations known a priori.

Type 2 uncertainty has been more widely addressed in the SP literature, perhaps due to one of its subclasses having a strong connection to exogenously uncertain problems. In the taxonomy presented in Hellemo et al. (2018), this specific type of endogenous uncertainty is called *conditional information revelation*. In this subclass, the decisions only affect the time at which the (exogenous) uncertainty is revealed to the decision maker. One of the earliest publications on such uncertainty is Jonsbråten et al. (1998), where the authors describe a branching algorithm for solving a subcontracting problem. Goel and Grossmann (2006) consider a process network problem where the yield of a new process is uncertain prior to installation. Other applications include open pit mining

(Boland et al., 2008), clinical trial planning for drug development (Colvin and Maravelias, 2010) and technology project portfolio management (Solak et al., 2008).

Similar solution methods are employed in problems with exogenous and Type 2 endogenous uncertainties. The main difference is that conditional information revelation requires the use of conditional non-anticipativity constraints (C-NACs), as the distinguishability is dependent on earlier decisions. This conditional dependency results in disjunctive constraints that require specific reformulation techniques (Apap and Grossmann, 2017). The main challenge arising from this approach is that the number of constraints rapidly increases with problem size, resulting in computational intractability for large problems. Apap and Grossmann (2017) also propose omitting redundant constraints, in an attempt to mitigate the tractability issues. In their example production planning problem, this results in roughly a 99% decrease in the problem size. Despite these improvements, the reduced model is still very large and cannot be solved to the optimum within a reasonable computation time under their experimental setting, illustrating how challenging such problems are.

In addition to information revelation, Type 2 uncertainty also encompasses problems where the information structure is altered by modifying the support of decision variables and changing the objective or constraint coefficients. Hellemo et al. (2018) reported a 2SSP problem where the recourse costs depend on first stage decisions (Ntaimo et al., 2012). Gustafsson and Salo (2005) present a model for contingent portfolio programming, a project scheduling problem where projects can be expanded or terminated before they are finished. The decisions thus affect the decision spaces of future decision variables, resulting in additional consistency constraints, e.g., that a project can be continued in period t only if it was ongoing in period $t-1$.

Type 1 endogenous uncertainty is more challenging from a mathematical modeling standpoint because the uncertain events depend on earlier decisions, which, in turn, precludes a scenario tree-based representation as the scenario probabilities in a tree cannot depend on decisions. Therefore, the well-established solution techniques for MSSP cannot be directly applied to these problems.

Despite these challenges, some discussion on Type 1 uncertainty is found in the literature. Peeta et al. (2010) discuss the fortification of a structure in a network, where the probability of failure depends on the fortification decision. Lauritzen and Nilsson (2001) present the "pig farm problem", where the health of a pig depends on the treatment decisions, introducing the concept of limited memory influence diagrams (LIMID), thus relaxing the no-forgetting assumption in influence diagrams. The no-forgetting assumption states that when making a decision, all prior decisions and outcomes of uncertain events are known. This assumption is a significant limitation in distributed decision-making and leads to computational challenges in multi-period problems where the decisions towards the end are conditional on the full history of the problem. Dupačová (2006) presents a summary of problems with Type 1 uncertainties. Finally, reformulations and custom algorithms for various Type 1 problems are further summarized in Hellemo et al. (2018). However, none of these approaches are easily generalizable to different problems.

In this paper, we present a general solution framework for problems with both types of endogenous uncertainty discussed here. Modeling Type 1 endogenous uncertainties has previously relied on specific problem structures and reformulations, and the ability to model these uncertainties in a general setting makes the proposed framework versatile compared to these earlier methods. Additionally, we present two different ideas for incorporating Type 2 endogenous uncertainty in the decision models formulated with our framework. The ease of considering any combination of the two types of endogenous uncertainty also makes the framework considerably more applicable than the approaches discussed earlier to treat endogenous uncertainty in general.

3. Decision programming with conditionally observed information and subdiagram decomposition

3.1. Decision Programming

Decision Programming relies on a graph representation of the problem, in which nodes represent decisions and the observation of chance events. In specific, let $G(N, A)$ be an acyclic graph formed by nodes $N = C \cup D \cup V$, where C and D are the sets of chance and decision nodes, respectively; V is a value node (or a collection of value nodes) representing the consequence(s) incurred from the decisions made at nodes D and the chance events realized at nodes C. Each decision and chance node $j \in C \cup D$ can assume a state s_j from a discrete and finite set of states S_j. For a decision node $j \in D$, S_j represents the available choices; for a chance node $j \in C$, S_j is the set of possible realizations. The set $A = \{(i, j) \mid i, j \in N\}$ contains the arcs (i, j), which represent the influence between nodes. Before defining this notion of influence further, let us first define a few necessary concepts.

The *information set* comprises the immediate predecessors of a given node $j \in N$, and is defined as $I(j) = \{i \in N \mid (i, j) \in A\}$. The decision made in each decision node and the conditional probabilities of the states in each chance node ($s_j \in S_j$, $j \in D$ and $j \in C$, respectively) depend on their *information state* $s_{I(j)} \in S_{I(j)}$, where $S_{I(j)} = \prod_{i \in I(j)} S_i$ is the set of all possible information states for node j.

Let us define $X_j \in S_j$ as the realized state at a chance node $j \in C$. For a decision node $j \in D$, let $Z_j : S_{I(j)} \mapsto S_j$ be a mapping between each information state $s_{I(j)} \in S_{I(j)}$ and decision $s_j \in S_j$. That is, $Z_j(s_{I(j)})$ defines a local decision strategy, which represents the choice of some $s_j \in S_j$ in $j \in D$, given the information $s_{I(j)}$. Such a mapping can be represented by an indicator function $\mathbb{I} : S_{I(j)} \times S_j \mapsto \{0, 1\}$ defined so that

$$\mathbb{I}(s_{I(j)}, s_j) = \begin{cases} 1, & \text{if } Z_j \text{ maps } s_{I(j)} \text{ to } s_j, \text{ i.e., } Z_j(s_{I(j)}) = s_j; \\ 0, & \text{otherwise.} \end{cases}$$

A (global) *decision strategy* is the collection of local decision strategies in all decision nodes: $Z = (Z_j)_{j \in D}$, selected from the set of all possible strategies $\mathbb{Z}$.

A *path* is a sequence of states $s = (s_i)_{i=1,\ldots,n}$, with $n = |C| + |D|$; and

$$S = \{(s_i)_{i=1,\ldots,n} \mid s_i \in S_i, i = 1, \ldots, n\} \tag{1}$$

is the set of all possible paths. Moreover, the information states $s_{I(j)}$ and states s_j can be seen as subpaths derived from a given path s and we say that a strategy Z is compatible with a path s if $Z_j(s_{I(j)}) = s_j$ for all $j \in D$.

Using the notion of information states, the conditional probability of observing a given state s_j for $j \in C$ is $\mathbb{P}(X_j = s_j \mid X_{I(j)} = s_{I(j)})$. The probability associated with a path s being observed given a strategy Z can then be expressed as

$$\mathbb{P}(s \mid Z) = \left(\prod_{j \in C} \mathbb{P}(X_j = s_j \mid X_{I(j)} = s_{I(j)}) \right) \left(\prod_{j \in D} \mathbb{I}(s_{I(j)}, s_j) \right) \tag{2}$$

Notice that the term $\prod_{j \in D} \mathbb{I}(s_{I(j)}, s_j)$ in equation (2) takes value one if the strategy Z is compatible with the path s, and zero otherwise. Thus, one can pre-calculate the probability

$$p(s) = \left(\prod_{j \in C} \mathbb{P}(X_j = s_j \mid X_{I(j)} = s_{I(j)}) \right).$$

of a path s being observed, given that the selected strategy is compatible with s.

At the value node $v \in V$, a real-valued utility function $U_v : S_{I(v)} \mapsto \mathbb{R}$ maps the information state $s_{I(v)}$ of v to a utility value U_v. The default objective is to maximize the expected utility of a strategy, which can be expressed as

$$\sum_{v \in V} \sum_{s \in S} \mathbb{P}(s \mid Z) U_v(s_{I(v)}) = \sum_{s \in S} \mathbb{P}(s \mid Z) \sum_{v \in V} U_v(s_{I(v)}).$$

A significant advantage of using Decision Programming is that other objectives such as conditional Value-at-Risk can also be used, as discussed in Salo et al. (2022).

This problem can be formulated as a mixed-integer linear programming (MILP) model, which allows for employing standard techniques widely available in off-the-shelf solvers. For that, let us define the binary variable $z(s_j \mid s_{I(j)})$ that takes value 1 if $Z_j(s_{I(j)}) = s_j$, and 0 otherwise, for all $j \in D$, $s_j \in S_j$, and $s_{I(j)} \in S_{I(j)}$. Then, the MILP problem can be stated as (3)-(8).

$$\max_{Z \in \mathbb{Z}} \sum_{s \in S} \pi(s) U(s) \tag{3}$$

$$\text{subject to} \sum_{s_j \in S_j} z(s_j \mid s_{I(j)}) = 1, \qquad \forall j \in D, s_{I(j)} \in S_{I(j)} \tag{4}$$

$$0 \leq \pi(s) \leq p(s), \qquad \forall s \in S \tag{5}$$

$$\pi(s) \leq z(s_j \mid s_{I(j)}), \qquad \forall j \in D, s \in S \tag{6}$$

$$\pi(s) \geq p(s) + \sum_{j \in D} z(s_j \mid s_{I(j)}) - |D|, \qquad \forall s \in S \tag{7}$$

$$z(s_j \mid s_{I(j)}) \in \{0, 1\}, \qquad \forall j \in D, s_j \in S_j, s_{I(j)} \in S_{I(j)}. \tag{8}$$

Variables $\pi(s)$ are nonnegative continuous variables representing the conditional path probability in Eq. (2). They take the value of the path probability $p(s)$ in case the selected strategy Z is compatible with the path s, and zero otherwise. Notice that this is equivalent to observing $z(s_j \mid s_{I(j)}) = 1$ for all $j \in D$.

The objective function (3) defines the expected utility value, which is calculated considering only the paths that are compatible with the strategy. We denote the total path utility as $U(s) = \sum_{v \in V} U_v(s_{I(v)})$. The correct behaviour of variables $\pi(s)$ is guaranteed by constraints (5)-(7), which enforce that $\pi(s) = p(s)$ if $z(s_j \mid s_{I(j)}) = 1$ for all $j \in D$. The term $|D|$ in (7) is the number of decision nodes in the diagram.

3.2. Conditionally observed information

As originally proposed, Decision Programming focuses on problems with Type 1 endogenous uncertainty, i.e., decision-dependent probabilities. However, many MSSP problems involve conditionally observed information (also called conditional information revelation). Before introducing further extensions, let us illustrate the notion of conditionally observed information with a capacity expansion problem described in Goel and Grossmann (2006). The setting is a chemical process network where a product is made and then delivered to customers. The decision maker (DM) can increase production by installing a new process with an uncertain efficiency that can be observed only after installing the machine.

A key concept with conditionally observed information is *distinguishability*. The potential efficiency of the new process, if installed, is independent of the DM's choice to install it. If that

efficiency differs between two paths s and s', the paths are different. However, from the DM's perspective, there is no difference between s and s' if the new process is not installed and the efficiency is thus not observed. Such paths are *indistinguishable* at that point in the decision process.

In conditionally observed information, the two key elements are the decisions or random events that the observation is conditional on, and the condition that the observation depends on. We refer to these as the *distinguishability set* $D_{i,j} \subset C \cup D$ and the *distinguishability condition* $F_{i,j} : S_{D_{i,j}} \mapsto \{0, 1\}$, respectively. Here, $i \in C \cup D$ denotes the conditionally observed node, usually a chance node, and $j \in D$ is the decision node where that information is available if the distinguishability condition is fulfilled.

Using the notion of distinguishability sets and conditions, we can define *conditional arcs*

$$a_c \in A_c = \{(i, j) \mid i \in C \cup D, j \in D, i < j,\ D_{i,j} \neq \emptyset,\ F_{i,j}(s_{D_{i,j}}) = 1\}. \tag{9}$$

to describe conditionally observed information in influence diagrams.

Specifically, we say that a conditional arc a_c from node $i \in C \cup D$ to node $j \in D$ is active (i.e., node i is observed when making the decision corresponding to node j) if the condition $F_{i,j}$ is fulfilled by the states of the nodes $k \in C \cup D$ in the distinguishability set $D_{i,j}$. If $D_{i,j}$ is empty, there is no conditional observation of i in j and, thus, no conditional arc between these nodes exists. The concept is illustrated in Fig. 1. We note that the conditional arc is somewhat analogous to a transistor, a common electronic component, which allows current to pass through if the correct control voltage is applied. Analogously, a conditional arc allows information to pass through if the correct states of the nodes in the distinguishability set are realized, i.e., $F_{i,j}(s_{D_{i,j}}) = 1$.

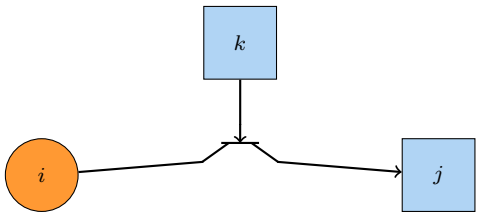


Figure 1: An illustration of a conditional arc (i, j). Nodes i and j correspond to the earlier notation where the realization of i is conditionally observed in j. The distinguishability set is $D_{i,j} = \{k\}$.

In the capacity expansion example from Goel and Grossmann (2006), the chance node i in Fig. 1 corresponds to the yield of the new process, decision node k to the installation decision, and j to the operational decisions such as which processes and raw materials to use for production. The distinguishability set is $D_{i,j} = \{k\}$, and the distinguishability condition is $F_{i,j}(s_{D_{i,j}}) = \mathbb{I}(s_k =$ "install new process"), where the indicator function $\mathbb{I}(\,\cdot\,)$ is defined as

$$\mathbb{I}(x = x^*) = \begin{cases} 1\,(true), & \text{if } x = x^*, \\ 0\,(false), & \text{otherwise.} \end{cases}$$

If the distinguishability set includes more than one node, alternative functions $F_{i,j}$ might be employed for modeling the conditional dependencies between the nodes. For example, if there are several projects that reveal the same information in node $i \in N$ and completing any of these projects is sufficient for information revelation, $F_{i,j}(s_{D_{i,j}}) = \bigvee_{k \in D_{i,j}} \mathbb{I}(s_k = s_k^*)$ can be used; or if all of the projects are required for the information revelation, $F_{i,j}(s_{D_{i,j}}) = \bigwedge_{k \in D_{i,j}} \mathbb{I}(s_k = s_k^*)$ is appropriate. An example of such conditions is found in Tarhan et al. (2009), where different uncertainties in oil field development are gradually revealed, and the uncertainty in the amount of recoverable oil in a reservoir can be resolved in two different ways, namely drilling a sufficient number of wells or using the reservoir for production for long enough.

3.3. Incorporating conditionally observed information in Decision Programming

The conditional arcs are designed to describe conditionally observed information in influence diagrams. However, they are a general representation of the concept, not a solution method. In what follows, we present two alternative approaches for incorporating this concept into the Decision Programming framework, which ultimately enables solving Type 3 endogenously uncertain stochastic problems. The first approach employs *observation nodes*, used in decision analysis problems such as the used car buyer problem (Howard and Matheson, 2005). The second approach utilizes *conditional non-anticipativity constraints*, which are used in stochastic programming for modeling the decision-dependent information structure.

Observation nodes portray how the decision maker observes the information, see Herrala (2020) for details. Earlier decisions affect the probability distribution of the observations, and thereby Type 2 uncertainty is effectively transformed into Type 1 uncertainty, making it directly amenable to a Decision Programming formulation. This approach is also used in Salo et al. (2022).

In effect, each conditional arc is replaced with an observation node, as illustrated in Fig. 2. The information set of the observation node is the union of the chance node i and the distinguishability set $D_{i,j}$ ($D_{i,j} = \{k\}$ in the example from Fig. 1), and the state space is $S_i \cup$ "no observation". Then, the observation node replaces the node $i \in C$ in the information set of $j \in D$, controlling whether or not the information in i is available in j. A benefit of this approach is that wrong or imperfect observations can also be modeled, e.g., if an expert could be consulted in the chemical production example to give an educated but nevertheless uncertain guess on the efficiency of the new process.

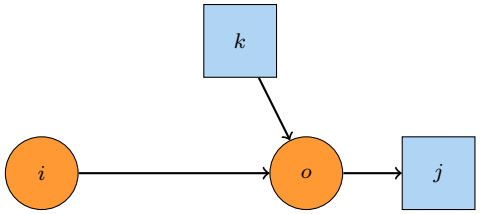


Figure 2: Replacing a distinguishability arc with an observation node in the example from Fig. 1.

One can also utilize the ideas of Ruszczyński (1997) and Apap and Grossmann (2017) on modeling conditional information revelation, in which the information structure can be connected to the decisions by using disjunctive constraints. Specifically, let us first define (in)distinguishability of scenarios. Two scenarios s and s' in a MSSP with $|T|$ stages are said to be indistinguishable at a stage (or time period) $t \in T$ if all observed realizations of the uncertain parameters until that stage are the same. In Decision Programming, the scenario paths also include the decisions as the probability structure can depend on them, and thus for any two paths to be indistinguishable, the decisions up to time $t \in \{1, \ldots, t_{max}\}$ (recall that a path $s \in S$ is defined as $(s_i)_{i=1,\ldots,n}$) must also be the same.

In stochastic programming, NACs state that the decisions at time t must be the same between indistinguishable scenarios. For problems with only exogenous uncertainty, defining NACs is relatively straightforward, since each uncertainty is necessarily observed at a predefined time period. In contrast, endogenous uncertainty requires conditional non-anticipativity constraints (C-NACs). C-NACs are disjunctive constraints (see Lee and Grossmann (2000) for a summary on disjunctive programming) based on the same ideas as the distinguishability arcs: unless some condition that makes two scenarios distinguishable at time $t \in T$ is fulfilled, the C-NAC is active and the decision must be the same in these scenarios. Instead of defining NACs for such pairs of decision variables

where the scenarios are known to be indistinguishable, C-NACs must be defined for all pairs where the scenarios *can be* indistinguishable, often leading to a large number of such constraints.

To integrate C-NACs into Decision Programming, the conditional arcs introduced earlier need to be supplemented with the *conditional information set* $I_c(j) = \{i \in C \cup D \mid (i,j) \in A_c\}$ to represent the conditionally available information at node $j \in D$. The C-NACs then control whether or not this information is available when making the decision in node j. We observe that two paths s and s' are always distinguishable at node $j \in D$ if the non-conditional information states $s_{I(j)}$ and $s'_{I(j)}$ are different. Instead, C-NACs are needed when the conditional information states differ.

If the conditional information states differ (i.e., node $i \in I_c(j)$ has different states in paths s and s'), distinguishability is dependent on the corresponding distinguishability condition(s) $F_{i,j}(s_{D_{i,j}})$. This distinguishability of two paths at node $j \in D$ can be formulated as a Boolean variable $f_j^{s,s'}$, defined as

$$f_j^{s,s'} = \bigvee_{i \in I_c(j) | s_i \neq s'_i} F_{i,j}(s_{D_{i,j}}).$$

The value of $f_j^{s,s'}$ is *True* if the conditionally revealed information makes scenarios s and s' distinguishable at node j, and *False* otherwise.

Finally, we extend the definition of the local decision strategy $Z_j(s_{I(j)})$ and the corresponding binary variables $z(s_j \mid s_{I(j)})$ to include the conditional information set $I_c(j)$ in order to simplify the notation. From a formulation standpoint, $I_c(j)$ is simply a part of the information set of a node, and C-NACs are used to enforce non-anticipativity of decisions, but we use this notation to emphasize the role of conditionally revealed information. Combining these ideas, we can define C-NACs in the context of Decision Programming as

$$\neg f_j^{s,s'} \implies Z_j(s_{I(j)}, s_{I_c(j)}) = Z_j(s_{I(j)}, s'_{I_c(j)}), \qquad \forall s_{I(j)} \in S_{I(j)},\, s_{I_c(j)}, s'_{I_c(j)} \in S_{I_c(j)} \tag{10}$$

$$\implies z(s_j \mid s_{I(j)}, s_{I_c(j)}) = z(s_j \mid s_{I(j)}, s'_{I_c(j)}), \quad \forall s_j \in S_j,\, \forall s_{I(j)} \in S_{I(j)},\, s_{I_c(j)}, s'_{I_c(j)} \in S_{I_c(j)}. \tag{11}$$

In light of the above, the C-NACs for binary variables z can also be conveniently written as

$$|z(s_j \mid s_{I(j)}, s_{I_c(j)}) - z(s_j \mid s_{I(j)}, s'_{I_c(j)})| \leq f_j^{s,s'},\, \forall s_j \in S_j,\, \forall s_{I(j)} \in S_{I(j)},\, s_{I_c(j)}, s'_{I_c(j)} \in S_{I_c(j)}. \tag{12}$$

Notice that the absolute value function used in the left-hand side of (12) can be trivially linearized without compromising the model complexity. The number of constraints (12) introduced for a decision node $j \in D$ can be as high as $(|S_j||S_{I(j)}|) \times (\prod_{i \in I_c(j)} |S_i|^2)$ if one C-NAC is introduced for every pair of states s_i and s'_i for every node i in the conditional information set $I_c(j)$. The first term $|S_j||S_{I(j)}|$ states that C-NACs must be defined for all states of the decision node j and its non-conditional information set $I(j)$. The second term $\prod_{i \in I_c(j)} |S_i|^2$ seems daunting for practical modeling, but can be decreased using ideas from Apap and Grossmann (2017). First, C-NACs obviously need not be introduced for pairs where $s_i = s'_i$. Second, due to symmetry, half of the remaining constraints are redundant because the constraint (12) is the same for state pairs (s_i, s'_i) and (s'_i, s_i). Additionally, we only need to consider cases where the state of exactly one conditionally revealed node is different between the two paths s and s' (see Apap and Grossmann (2017) for further explanation). With these reductions, the number of constraints is greatly decreased, increasing the computational feasibility of this approach.

With these C-NAC reduction properties, representing the decision-dependent information structure might be more compact than the corresponding model using observation nodes. However, C-NACs lack the ability to model Type 1 endogenous uncertainty. Observation nodes are more

versatile, allowing for modeling both types of endogenous uncertainty within one observation node, making it possible to model and solve T3ESP problems. In Section 3.5, we consider both modeling approaches and compare their applicability and computational requirements in different settings, and in Section 4, we present a large-scale example problem involving different endogenous uncertainties.

3.4. Subdiagram decomposition

A well-known challenge in decision analysis and stochastic programming is the *curse of dimensionality* (Bellman, 1961), which describes the characteristic exponential growth of problems as the number of nodes or stages increases. To mitigate this exponential growth, we propose a decomposition approach for Decision Programming. In general, decomposition splits the problem into a main problem and subproblem(s). Methods like Benders (Benders, 1962) and Dantzig-Wolfe (Dantzig and Wolfe, 1960) decomposition use this idea to make large problems computationally tractable. Both methods require a specific problem structure, which is then exploited in a divide-and-conquer strategy. The subproblems are smaller, independent parts of the full problem. If the problem is separable, the solutions of the subproblems can be utilized in the main problem, which recombines them into a solution for the full problem. After introducing an illustrative example, we present conditions that exploit the memory of the decision maker to ensure separability of Decision Programming problems.

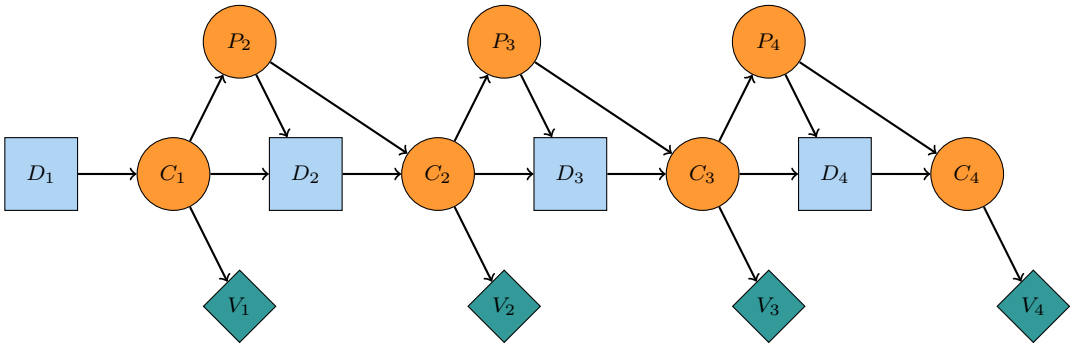


Figure 3: A four-period version of the unfair coins problem.

To demonstrate the decomposition, we introduce a multi-period "unfair coins" game presented in Fig. 3. In every period, the player makes a decision of whether play or skip the turn, represented by nodes D_i. If the player decides to play, they might win or lose money. If they skip, no money is gained or lost. This outcome is represented by nodes C_i and the associated value by V_i. In each period, a possibly unfair coin is randomly chosen from three alternatives in P_i. The probabilities of picking different coins, i.e., the conditional probability distributions in P_i, depend on the previous period outcome C_{i-1}. The arcs from P_i to D_i present the assumption that the player always observes which coin was chosen before making the decisions, thus knowing the probability of winning each round. If the player wins a given round, the highest winning probability coin is more likely to be chosen in the next period, and losing similarly makes the lower winning probability coin more likely. Skipping a turn makes the "middle" coin the most likely.

The subdiagram decomposition we propose is based on splitting the influence diagram into two subdiagrams, which represent the main and subproblems. The main problem formulation is almost identical to (3)-(8), with the full path s replaced by a main problem path s_{main} denoting the states of the main problem nodes N_{main}, and the full decision strategies Z replaced by Z_{main}, consisting of local decision strategies $Z_j(s_{I(j)})$ for each main problem decision node $j \in D_{main} = \{j \mid j \in$

$N_{main}, j \in D\}$. We assume an arbitrary utility function $U(s) = U(s_{main}, s_{sub})$ and an expected value objective function for the full problem. In practice, we solve a subproblem and obtain an expected utility for each subpath $s_{main} \in S_{main}$.

Adding a stage to the game introduces two chance nodes and a decision node, rapidly increasing the number of paths in the full problem. To decrease solution times (computational experiments can be found in Section 3.5), we use the subdiagram decomposition and split the problem by periods so that all nodes in periods 3 and onwards, for example, are moved into the subproblem.

The first condition required for separability is that all main problem nodes must be in the information set of each subproblem decision node:

$$N_{main} \subseteq \{i \mid i \in I(j),\ \forall j \in D_{sub}\}.$$

To motivate this partitioning, we note that if there was a node $i \in N_{main}$ not belonging to the information set of a subproblem decision node $j \in D_{sub}$, the decision in each subproblem could be different for different states $s_i \in S_i$ as the decomposition would effectively add i to the information set of each subproblem decision node. In our example problem (Fig. 3), we assume that all previous decisions and realizations are remembered when making a decision, but omit these arcs from the influence diagram for clarity. The first condition is thus fulfilled by this influence diagram. If the node P_3 was not remembered when making the decision D_3, a decomposition where P_3 remained in the main problem and D_3 was moved to the subproblem would implicitly add P_3 to $I(D_3)$, and the decomposed problem would no longer be equivalent to the full problem.

The utility $U(s_{main})$ is the optimal solution to the subproblem

$$U(s_{main}) = \max_{Z_{sub} \in \mathbb{Z}_{sub}} \sum_{s_{sub} \in S_{sub}} \pi(s_{sub} \mid s_{main}) U(s_{main}, s_{sub}) \tag{13}$$

$$\text{subject to} \sum_{s_j \in S_j} z(s_j \mid s_{I(j)}) = 1, \qquad \forall j \in D_{sub}, s_{I(j)} \in S_{I(j)} \tag{14}$$

$$0 \leq \pi(s_{sub} \mid s_{main}) \leq p(s_{sub} \mid s_{main}), \qquad \forall s_{sub} \in S_{sub} \tag{15}$$

$$\pi(s_{sub} \mid s_{main}) \leq z(s_j \mid s_{I(j)}), \qquad \forall j \in D_{sub}, s_{sub} \in S_{sub} \tag{16}$$

$$\pi(s_{sub} \mid s_{main}) \geq p(s_{sub} \mid s_{main}) + \sum_{j \in D_{sub}} z(s_j \mid s_{I(j)}) - \mid D_{sub} \mid, \quad \forall s_{sub} \in S_{sub} \tag{17}$$

$$z(s_j \mid s_{I(j)}) \in \{0, 1\}, \qquad \forall j \in D_{sub}, s_j \in S_j, s_{I(j)} \in S_{I(j)}. \tag{18}$$

If we multiply all the conditional probabilities together appropriately, we can see that the decomposed objective function

$$\max_{Z_{main} \in \mathbb{Z}_{main}} \sum_{s_{main} \in S_{main}} \pi(s_{main}) U(s_{main})$$

is equivalent to the objective function (3).

However, this is not sufficient for the decomposed problem to be equivalent to (3)-(8). To ensure separability and the correct flow of information, we impose a further condition that the states of subproblem nodes cannot influence the main problem: $\forall i \in N_{sub} : i \notin I(j),\ \forall j \in N_{main}$. In some problems, manipulating the influence diagram with arc reversals, as discussed in Shachter (1986), can be used to achieve this using conditional expectation and Bayes' rule. Note that arcs from chance nodes to decision nodes cannot be reversed, because doing so would require converting the decision node j into a chance node representing the local decision strategy $Z_j(s_{I(j)})$. However, this decision strategy is determined by the values of the z-variables in the main problem, which in turn

depend on the subproblem solutions. This interdependence means that neither problem can be solved before the other, and the decomposition cannot be applied.

To summarize, with an arbitrary path utility function, the proposed decomposition requires the following two conditions to hold:

1. All main problem nodes must be in the information set of each subproblem decision node.
2. No subproblem node can be in the information set of a main problem node.

After calculating the subproblem utilities for all $s_{main} \in S_{main}$, we obtain a main problem strategy that also maximizes the subproblem utilities. When deciding the optimal strategies in the subproblem, the conditions we have proposed here allow for exploiting the memory of the decision maker(s) to represent the main problem strategies and thus creating a valid decomposition. The constraints (14)-(18) ensure that the z-variables in the subproblem represent local strategies $Z_j(s_{I(j)})$ and the π-variables are the probabilities of each subproblem path given the main problem path and subproblem decision strategy. With the proposed conditions, no information is lost in the decomposition and the main problem path is sufficient for determining the corresponding optimal subproblem strategy.

While being otherwise a versatile framework, Decision Programming is strictly limited to problems in which both the decision spaces S_j of decision nodes $j \in D$ and the sets of possible outcomes S_j for the chance nodes $j \in C$ are discrete and finite. This imposes a limitation on practical applications that benefit from the employment of mathematical programming as a modeling framework. To circumvent the requirement of discrete and finite state spaces, we note that if there is no Type 1 endogenous uncertainty in the subproblems, they can be represented with other optimization approaches, thereby allowing for continuous decision variables. This is demonstrated in Section 4.

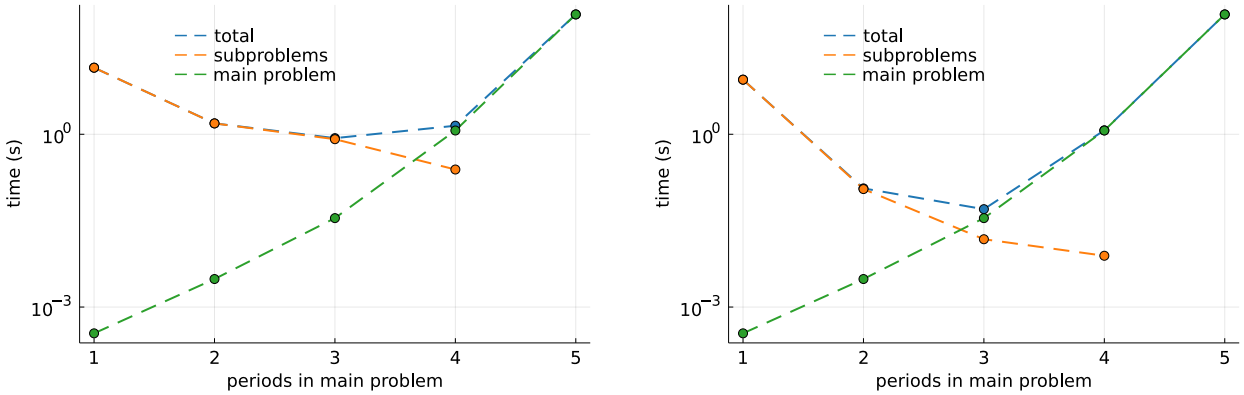
The proposed framework also exposes a natural separation between the evaluation of the expected utility functions of subproblems and the solution of the main problem. This separation can be thus exploited in a decomposed fashion and solving the subproblems greatly benefits from the employment of parallelized solution strategies. This creates a possibility to trade off the computational demand between evaluating the subproblems and solving the main problem. For example, once a node $j \in D \cup C$ is moved into the subproblem, the total number of paths $|S|$ in the main problem is reduced by a factor of $|S_j|$, shifting some of the computational burden from solving the main problem to solving a collection of subproblems, which are in turn more demanding than calculating the path utilities $U(s)$ in the corresponding full problem (3)-(8). In Section 4, we demonstrate exploiting this trade-off, making a complex problem computationally tractable by employing the proposed decomposition approach.

3.5. Computational experiments

The computational tractability of the framework is first tested with the game described in Section 3.4. Information about the computational setup and available source code for all computational experiments can be found in Appendix A. We use a five-period version of the "unfair coins" game and decompose the problem so that the first n_{main} periods are the main problem and the rest is moved into the subproblem. We solve 20 replications of the problem with random winning probabilities and present the average solution times for different values of n_{main} in Fig. 4. When $n_{main} = 5$, the main problem becomes the full problem and the top right corner of the figure represents the solution time without decomposition. The main problem solution time increases exponentially with the number of stages, and the subproblem solution times decrease similarly. The solution time of a single subproblem is similar to that of a main problem with the same number of periods, but the number of subproblems increases exponentially with the number of periods in the main problem and this makes the total subproblem solution time higher. However, the subproblems could also

be solved in parallel and, with sufficient parallel computing units, the total solution time would approach the sum of the main problem solution time and the solution time of the computationally most demanding subproblem, as shown in Fig. 4b.

The total solution time with or without parallelization is the lowest with three stages in the main problem and the last two stages in the subproblem. This supports the intuitive idea that the optimal split would have the main problem only slightly larger, and consequently more computationally demanding, than each subproblem, thus compensating for the possibly large number of subproblems. The difference in total solution time between the full problem and a decomposed problem with three periods in the main problem is two orders of magnitude with no parallelization in solving the subproblems. If all subproblems were solved in parallel, the difference would be three orders of magnitude instead.



(a) Subproblems solved in series

(b) Subproblems solved in parallel

Figure 4: The solution times of the main and subproblems for a five-period unfair coins problem with different splits of the problem. Notice the logarithmic y-axis.

We now introduce a modified example with conditionally observed information in Fig. 5. Instead of always observing the coin used in each round (represented by nodes P_i), the decision maker can choose to observe it, but with a cost. With this modification, the uncertainty in what coin will be picked in P_i depends on the previous decision D_{i-1} through C_{i-1} and observing this uncertainty is conditional on the decision D_i^o, making this a Type-3 endogenously uncertain SP (T3ESP) problem.

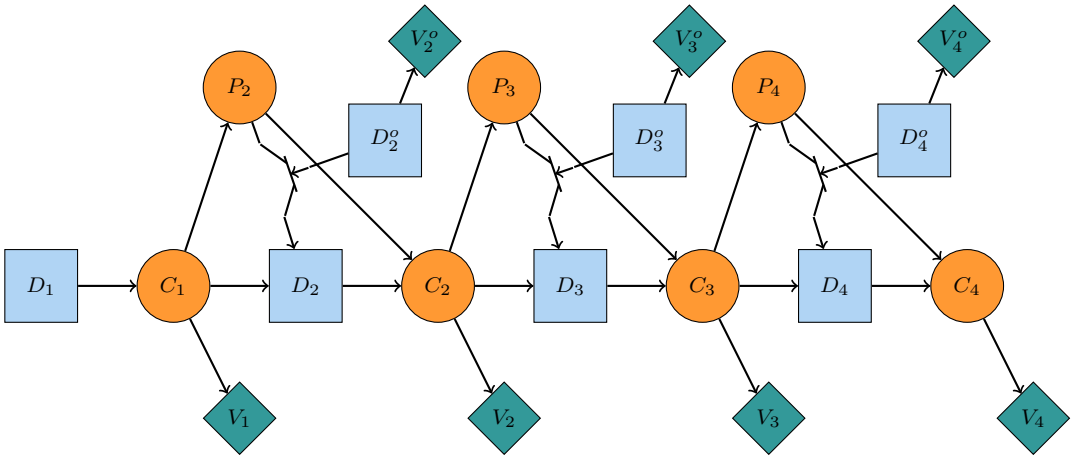


Figure 5: The game from Section 3.4 with conditionally observed information.

The conditional arcs can be modeled using observation nodes or conditional NACs as described in Section 3.3. The mean solution times for different problem sizes over 20 replications are presented in Fig. 6. First, we observe that making the information conditionally observed instead of always observed significantly increases solution times. This is explained by the additional decision of whether to observe the coin in each stage, which increases the problem size. There is practically no difference between solution times for observation nodes and C-NACs. Upon closer inspection of the solution process, we find that the two models are the same after the presolve process of the Gurobi solver. This is an interesting observation since it suggests that the solver is able to "see" the structure in conditionally observed information. Without presolve, solving the models takes significantly longer, but there are still no considerable differences between the solution times for observation nodes and C-NACs.

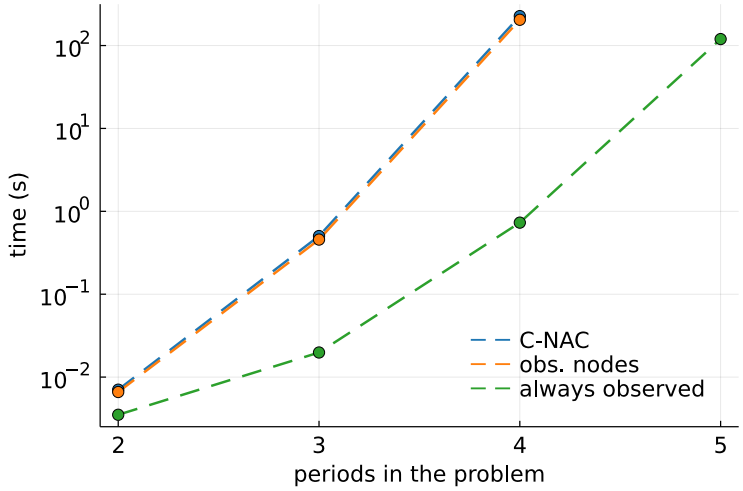


Figure 6: The solution times for different problem sizes using observation nodes or C-NACs. Notice the logarithmic y-axis.

In practice, both approaches have their advantages and disadvantages. Adding observation nodes only requires modifying the influence diagram by adding new nodes, while C-NACs are additional constraints that must be added to the decision model (3)-(8). One of the ideas behind Decision Programming is to provide a methodology for converting an influence diagram representation of the problem into a mixed-integer linear programming model, and from a usability perspective, it might be beneficial to not require explicit modification of the decision model. However, if one starts modifying the decision problem by using C-NACs, it might be possible to also represent other parts of the decision process explicitly as added variables or constraints. Further research in this direction is deemed out of scope for this paper.

A major disadvantage of observation nodes is that by the definition in Eq. (1), adding nodes to the diagram increases the number of paths $s \in S$. However, this does not necessarily increase the size of the decision model. To show this, we start with the definition that an observation node j is a deterministic mapping from an information state $s_{I(j)}$ to a state s_j, meaning that $\mathbb{P}(X_j = s_j \mid X_{I(j)} = s_{I(j)}) = 1$ for one of the states $s_j \in S_j$ and zero for others. From this, we observe that adding an observation node does not increase the number of paths for which the upper bound $p(s)$ in Eq. (5) is nonzero. For any path s with $p(s) = 0$, constraint (5) forces $\pi(s) = 0$ and the constraints (6)-(7) become redundant. It is therefore not necessary to consider such paths in the model and the problem size does not increase with added observation nodes. Nevertheless, the increased number of paths might prove challenging in implementing the model, since basic

operations such as iterating over all paths can become computationally infeasible. For example, the number of paths in the diagram in Fig. 5 is 17.9 million assuming three coins in each period.

4. Cost-benefit analysis for climate change mitigation

4.1. Model description

We now consider the cost-benefit analysis on mitigating climate change under uncertainty (see, e.g., Ekholm, 2018). Climate change is driven by greenhouse gas (GHG) emissions and can be mitigated by reducing these emissions, which incurs costs. However, mitigation reduces the negative impacts of climate change, referred to as climate damages. In cost-benefit analysis, the objective is to minimize the discounted sum of mitigation costs and climate damages over a long time horizon.

However, a few salient uncertainties complicate this calculation. First, it is not accurately known how much the climate will warm as a response to the rising GHG concentrations in the atmosphere, which is often expressed as climate sensitivity. Second, the severity of climate damages resulting from a given level of warming is uncertain. Earlier analyses (e.g. Ekholm, 2018; Ekholm and Baker, 2022) have assumed that these uncertainties are resolved exogenously over time. Moreover, additional uncertainty concerns the future progress of mitigation technologies (Rogelj et al., 2013), i.e., how much emission reductions will cost in the future.

Here, we model these three uncertainties endogenously, subject to deliberate research and development (R&D) efforts. The presented model is an extension from Ekholm and Baker (2022), which in turn is a simplification from the SCORE model (Ekholm, 2018). Both of these earlier versions assumed gradual and predetermined information revelation. At each time period, more accurate estimates of climate sensitivity and damages are obtained, and decisions on greenhouse gas reductions are made. Expanding on the previous works, we consider that the different R&D projects can resolve the uncertainties, but with uncertain outcomes, thus making the uncertainties endogenous. The resulting model is a multi-stage mixed-integer nonlinear problem (MINLP) with endogenous uncertainty. The state of technology in the future involves decision-dependent probabilities (Type 1), while the climate parameters are conditionally observed (Type 2).

The influence diagram for the problem is presented in Fig. 7. The first stage involves R&D decisions towards climate sensitivity (D_{CS}), damages (D_{Dmg}) and technology (D_{T1}). If successful, the climate parameter (climate sensitivity and damage exponent) R&D efforts modify the information structure so that the parametrization is partially revealed in 2050 instead of 2070. This is represented by the observation nodes O_{Dmg} and O_{CS} and the outcome (success/failure) of the projects by C_{Dmg} and C_{CS}.

Decisions over emission reductions ($D_{Ei}, i \in \{1, 2, 3\}$) are made in three stages: in 2030, 2050 and 2070, which represent the medium-term and long-term climate actions. The technological R&D potentially lowers the costs of emission reductions in 2050 and 2070. We connect our example to the discussion on the feasibility of large-scale deployment of bioenergy with carbon capture and storage (BECCS), which has been a crucial but contested result of many mitigation scenarios (Calvin et al., 2021).

The technology R&D decision in 2020 (D_{T1}) chooses between low or medium R&D effort. If the medium effort is chosen, one observes whether the R&D looks promising or not (C_{T1}), and can then decide in 2030 whether to continue with the medium R&D or switch to a higher level of R&D (D_{T2}). These observations are assumed to have no effect on the mitigation costs in 2030. Finally, C_{T2} represents the R&D outcome: low, medium or high level of mitigation costs in 2050 and 2070. In the "promising" scenario in C_{T1}, the low cost level is 50% more likely than originally perceived, and the high cost level is 50% less likely. For the "not promising" scenario, these effects are reversed. The R&D costs and probabilities for the three levels of BECCS costs are

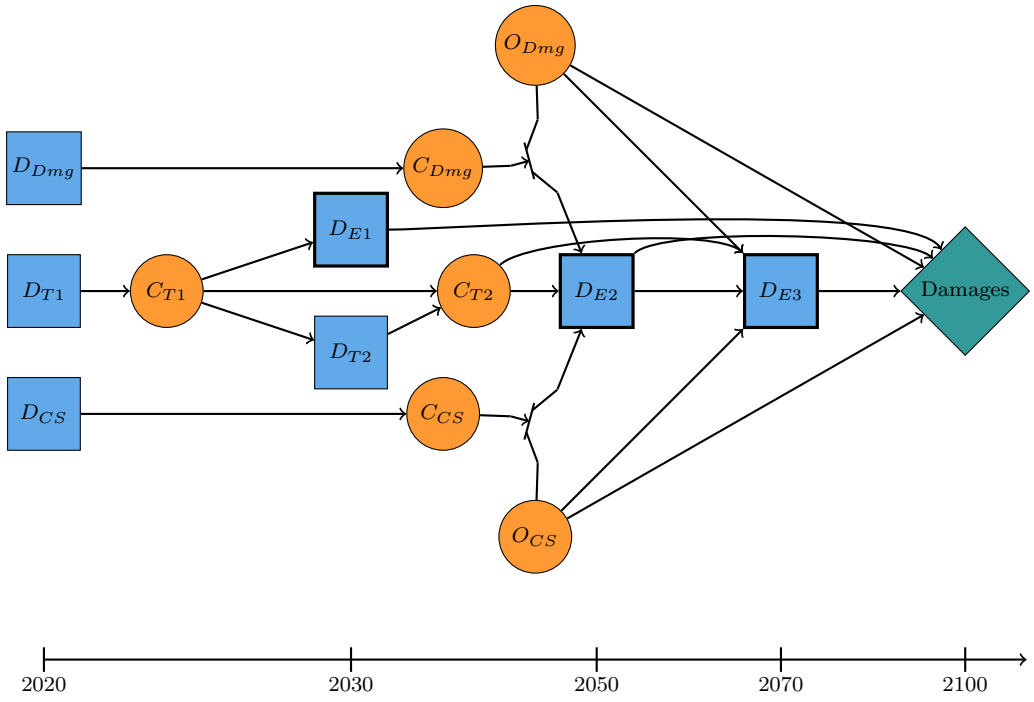


Figure 7: Influence diagram of the climate change cost-benefit problem with endogenous probability due to R&D. We assume that all prior decisions and uncertainty realizations apart from the conditionally observed parameters are remembered when making decisions, but omit the arcs for clarity.

parametrized using expert elicited estimates by Baker et al. (2015). These are then reflected in the overall emission reduction costs. It is worth noting the major challenges in long-term technological foresight, which is manifested in the wide spectrum of responses from the experts; but elicitation data is nevertheless useful for illustrating the importance of technological progress.

After 2070, the level of climate change is observed based on the chosen emission reductions and the observed branch of climate sensitivity, which then determines the climate damages along with the observed branch of climate damages.

4.2. Modifying the influence diagram

As discussed in Section 3, Decision Programming is limited to problems with discrete and finite state spaces for all nodes. However, discretizing the emission levels $D_{Ei}, i \in \{1, 2, 3\}$ would result in suboptimal solutions and we would thus prefer having continuous decision variables for the emission levels. To achieve this, we employ the decomposition presented in Section 3.4 and move the nodes $D_{Ei}, i \in \{1, 2, 3\}$ to the subproblem. Additionally, we move into the subproblem all nodes that are not in the information set of each of the emission level decisions. We assume that the previous decisions and chance node realizations are remembered in each decision node, but omit these arcs in Fig. 7 for clarity.

As a result, there is no Type 1 endogenous uncertainty in the subproblem and it can be modeled as a nonlinear three-stage stochastic programming problem with continuous decision variables. The nonlinearity comes from the SCORE model described in Appendix B. The impact of the decomposition on computational performance is discussed in Section 4.4.

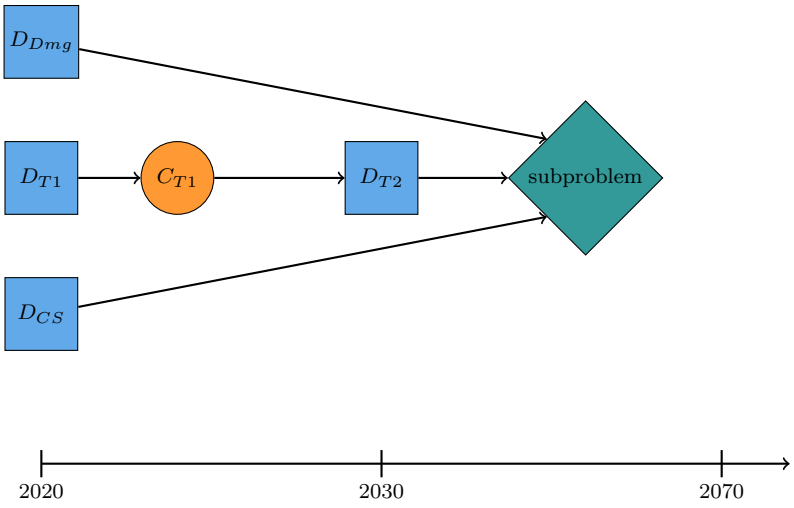


Figure 8: Influence diagram of the climate change cost-benefit main problem after decomposition.

4.3. Model results

The optimal R&D strategy for this problem is to carry out all R&D projects. We next discuss the endogenous impacts of the R&D outcomes in this optimal strategy.

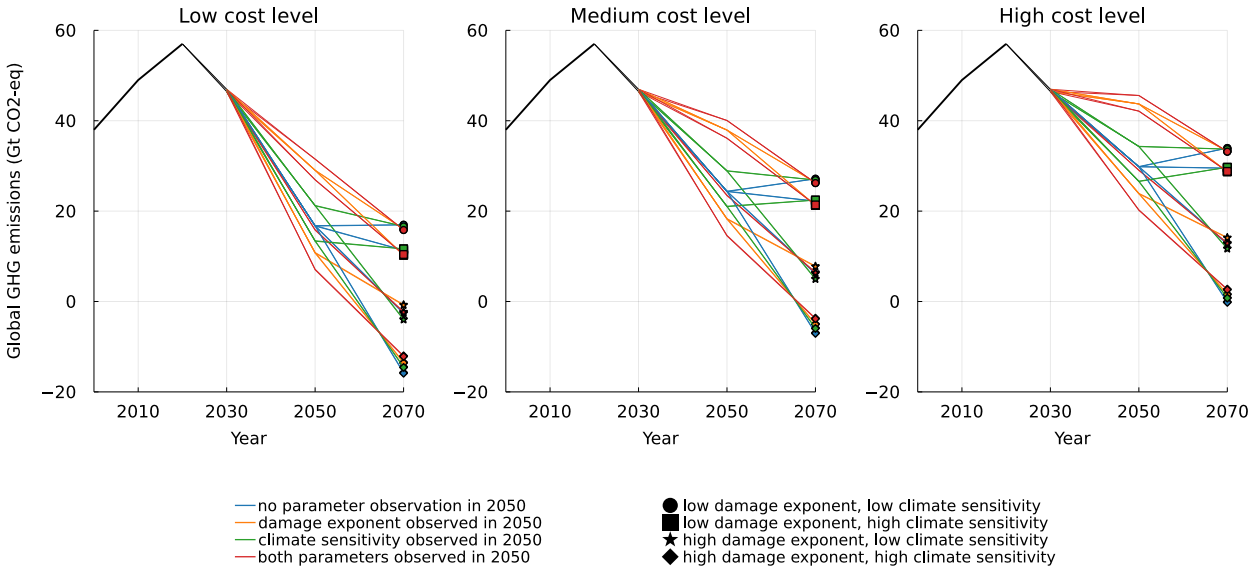


Figure 9: Abatement levels for different outcomes of technology research and parameter observations in 2030.

Fig. 9 presents the emission levels of the optimal mitigation strategy. The effect of the technology R&D on optimal emission pathways is shown with the three subfigures corresponding to the final R&D outcome after 2030. Intuitively, successful research and, therefore, cheaper abatement leads to more abatement. This has also a major impact on the total costs of the optimal strategy: with the low cost curves, the total expected cost is roughly 30% lower than with medium costs, and with high abatement costs 30% higher than with the medium cost curve.

The climatic parameter R&D, the other endogenous effect in this model has a magnitude smaller effect on the expected costs than the technology R&D, but the effect on abatement levels is remark-

able. The branches after 2030 represent different realizations of the technology research and partial learning of the climate parameters. If the research efforts for both parameters fail (blue lines in Fig. 9), the 2050 abatement decisions are made knowing only the outcome of the technology projects, and the four scenarios of partial learning only occur after 2050. Learning the parameters before 2050 results in more dispersed abatement strategies. Finally, the underlying parameter branching is represented with shapes in 2070. It can be seen that the impact of climate sensitivity is considerably smaller than that of the damage exponent. This is in line with the results in Ekholm and Baker (2022).

4.4. Computational aspects

Using the decomposed model presented in Fig. 8, the Decision Programming problem is solved in a tenth of a second, while solving all of the subproblems takes less than a minute. If we instead solve a problem corresponding to Fig. 7 without decomposition, the properties of the framework require discretizing the abatement levels. Using discrete decision spaces to approximate a continuous variable leads to suboptimal solutions. Furthermore, even with only three levels for each abatement decision, the problem becomes computationally very demanding. The path utilities are calculated in one second, but solving the Decision Programming model takes over an hour. With more abatement levels, the discretized problem would quickly become completely intractable. The decomposed problem instead uses continuous decision variables to represent the abatement decisions, precluding the need for a discretization. As a consequence, it is much faster to solve than the rudimentary approximation with three abatement levels.

The discretized model has 144876 constraints and 16662 variables, of which 138 are binary; and the decomposed main problem has 158 constraints and 36 variables, of which 12 are binary. The discretized model is thus three orders of magnitude larger than the decomposed model and the difference in solution times is even larger, but the path utilities are calculated almost instantaneously. However, the trade-off of moving some of the computational burden into the subproblems makes it possible to solve the decomposed problem to optimality. The discretized model can technically also be solved to optimality, but it requires a very rough discretization of the abatement levels, which are continuous decisions by nature. Even with a low number of abatement options, the solution times are very high compared to the decomposed model, and a low number of options is likely to result in solutions that are far from the optimal values. While discretization of decision spaces might seem like an obvious approach, as it enables direct implementation using the Decision Programming formulation (3)-(8), it is not computationally viable in all cases. Decomposition, in the cases where applicable, can provide better solutions with much smaller solution times.

5. Conclusion

In this paper, we have extended the Decision Programming framework (Salo et al., 2022), originally developed for solving decision problems with decision-dependent uncertainties by converting an influence diagram representation of the problem to a MILP. These extensions consist of a subdiagram decomposition approach to improve computational efficiency and allow for the inclusion of continuous decision variables, and two different approaches to consider conditionally observed information, making the framework more generally applicable to decision-making problems under uncertainty.

According to Hellemo et al. (2018), Type 3 endogenously uncertain stochastic programming (T3ESP) problems have not been discussed in the literature before. To make Decision Programming applicable to such problems, we show how Type 2 endogenous uncertainties can be modeled by

either adding observation nodes to the influence diagram (Herrala, 2020) or by adding conditional non-anticipativity constraints (C-NACs) to the Decision Programming model.

The computational experiments do not indicate clear superiority for either of these approaches, as the solution times observed were similar for both approaches in the example considered. However, in the presence of decision-dependent probability distributions, for example, the presence of imperfect conditional observations, one cannot employ C-NAC constraints. An example of such a problem would be a version of the climate CBA problem in Section 4 where the climate parameter research does not reveal the correct branch, that is, removes one of the extreme parameter values, but gives a probability distribution that provides better information than the original.

As with integer programming in general, the models rapidly grow in complexity when the model size increases. In our computational examples, this is clearly seen as solution times grow exponentially with the number of stages in the unfair coins game. Similar challenges can be observed in the climate change mitigation problem, where the full problem with discretized abatement decisions is computationally very demanding.

We observe that some problems are decomposable in a way that a part of the influence diagram can be converted into a subproblem. This allows for modeling a wider variety of problems as we are no longer strictly limited to discrete and finite decision spaces when the subproblems contain no decision-dependent probabilities. Furthermore, the same decomposition can decrease the total solution time by multiple orders of magnitude. In the climate change mitigation example in Section 4, decomposition makes the problem tractable and enabled obtaining more accurate solutions than with discretized abatement decisions.

In addition to the decomposition approach presented in this paper, the computational efficiency of Decision Programming could be vastly improved through other approaches. Ideas in this direction include decomposition methods for solving the MILP formulation (3)-(8), solution heuristics and tighter formulations. Incorporating new problem structures would make the framework applicable to a broader set of problems. Interesting examples of such research ideas are (distributionally) robust optimization and further examination of multi-objective decision-making, as discussed in Salo et al. (2022).

Despite the computational challenges and the number of possible improvements and extensions discussed, the framework is already general enough to model a challenging example problem with Type 3 endogenous uncertainty. It should also be noted that the problem formulation remains linear even with this type of uncertainty, guaranteeing global optimality of solutions.

Acknowledgements

Funding: The work of Herrala and Oliveira was supported by the Academy of Finland [*Decision Programming: A Stochastic Optimization Framework for Multi-Stage Decision Problems*, grant number 332180]; the work of Ekholm was supported by the Academy of Finland [*Designing robust climate strategies with Negative Emission Technologies under deep uncertainties and risk accumulation (NETS)*, grant number 331764].

Appendix A. Computational environment

All problems are solved using a desktop computer with an Intel i5-4690K CPU at 3.9GHz and 16GB of RAM, and the problem code was implemented in Julia v1.7.2 (Bezanson et al., 2017) using DecisionProgramming.jl v0.1.0 (Oliveira et al., 2021) with the Gurobi solver v9.5.1 (Gurobi Optimization, LLC, 2022) and JuMP v0.22.3 (Dunning et al., 2017). All the code used in the computational experiments is available and documented at a GitHub repository (Herrala, 2022).

Appendix B. Cost-benefit model description

The idea in climate change cost-benefit analysis (CBA) is the minimization of costs from emission abatement and climate damages. The abatement cost calculation is based on marginal abatement cost curves, as presented in equation (B.1), using numerical estimates from the SCORE model (Ekholm, 2018). Coefficients α and β are the parameters of the cost curves, R is the total abatement level and c is the marginal cost of abatement. In (B.3), C is the total cost for abatement level R. The subscript t has been omitted for clarity, but parameters α and β change between stages due to assumed technological progress.

$$R = \alpha c^{\beta} \tag{B.1}$$

$$\implies c = \left(\frac{R}{\alpha}\right)^{1/\beta} \tag{B.2}$$

$$\implies C = \int_0^R \left(\frac{r}{\alpha}\right)^{1/\beta} dr = \frac{\beta}{1+\beta}\left(\frac{1}{\alpha}\right)^{1/\beta} R^{1+1/\beta} \tag{B.3}$$

Departing from the predetermined technological progress that was assumed by Ekholm (2018), the parameter α in the model depends here on the result of technological R&D, as presented in Figure 7. We consider three levels of R&D effort in 2020 and 2030, which can then lead to three possible levels of MAC curves for years 2050 and 2070.

For the effect of R&D efforts on bioenergy and carbon capture and storage (CCS) costs, we used expert elicited estimates from Baker et al. (2015). To convert these into emission reduction costs, we assume a coal power plant as a baseline and calculate the additional costs from bioenergy with carbon capture and storage (BECCS) relative to the amount of reduced emissions by switching from coal to BECCS. Both plants were assumed to have a lifetime of 30 years and operate at 80% capacity on average. The coal power plant was assumed to have a 40% efficiency and produce 885 tonnes of CO_2 per GWh of electricity. The generation cost was assumed to be 50$/MWh. Costs were discounted at 5% rate.

Compared to coal, BECCS accrues additional costs per generated unit of electricity from the higher cost of biofuel and lower efficiency, and the additional investment to CCS and loss of efficiency from using some of the generated electricity in the carbon capture process. Baker et al. (2015) presented probability distributions for these parameters following three different levels of R&D efforts (low, medium and high). To calculate the cost differential to coal power plant, we did a Monte Carlo sampling of these four parameters, separately for each R&D level, which was then compared to the amount of reduced emissions per generated unit of electricity. This yields a distribution of emission reduction costs for BECCS for each R&D level.

We generalize the impact of R&D on BECCS's emission reduction cost to the overall marginal abatement cost (MAC) curve. This is obviously a simplification, but nevertheless reflects the major role that BECCS might have in decarbonizing the economy (e.g. Fuss et al., 2018; Rogelj et al., 2018). We take the high MAC from Ekholm (2018) as the starting point and define two MAC curves that are proportionally scaled down from the high MAC.

The low R&D level yields an average emission reduction cost around 100 $/t. We set three bins for three cost levels: high costs correspond to above 75 $/t, medium are between 25 and 75 $/t, and low costs below 25 $/t. The probabilities of achieving high, medium or low emission reduction costs are then estimated from the Monte Carlo sampling for each R&D level, and presented in Table B.1. With medium and high R&D effort, the average cost in the low cost bin is around 20 $/t. Therefore we assign the reduction in the MAC as 50% for medium costs and 80% for low

costs. The corresponding parameters are listed in Table B.2. These are still within the range of costs used in Ekholm (2018), where the low-cost MAC yielded the same emission reductions than the high-cost MAC with approximately 90% lower costs in 2050.

Table B.1: Probabilities for different abatement costs (rows) with different levels of R&D effort (columns).

	Low R&D	Medium R&D	High R&D
High costs	73 %	31 %	9 %
Medium costs	27 %	64 %	73 %
Low costs	0 %	5 %	18 %

Table B.2: Cost curve parametrization

year	α_{high}	α_{medium}	α_{low}	β
2030	3.57	3.57	3.57	0.340
2050	11.2	13.3	16.7	0.250
2070	21.1	24.3	29.3	0.203

The climate damage cost calculation is from DICE (Nordhaus, 2017). The damage function is presented in (B.4), where $Y(t)$ is the world gross economic output at time t, a is a scaling parameter and b is the damage exponent. While climate change and the abatement decisions have an effect on the economic output, the effect is assumed small and $Y(t)$ is defined exogenously in SCORE.

$$D(t, \Delta T) = Y(t) a \Delta T^b. \tag{B.4}$$

Finally, the temperature change ΔT is approximated with (B.5), where c is the climate sensitivity (the temperature increase from doubling of CO_2 emissions), M is the sum of emissions in 2030-2070 and k_i are coefficients.

$$\Delta T = k_1 c M + k_2 c + k_3 M + k_4. \tag{B.5}$$

In SCORE, both the DICE damage parameter and the climate sensitivity are uncertain with three options, low, medium and high, as presented in Table B.3, and the uncertainty is revealed in two steps in a binomial lattice. First, between 2050 and 2070, one of the extreme alternatives is removed from both uncertainties, that is, for both parameters, we know either that the value is not high or that it is not low. Then, after 2070, we learn the actual value.

The implementation here combines Decision Programming and MSSP in a way that the underlying branching probabilities are used as the probabilities of the observations O_{Dmg} and O_{CS} in Fig. 7. For the damage exponent, all branching probabilities are 50%. The observation O_{Dmg} thus has a 50% probability of removing either the high or low value. Depending on the branch, the low or high value then has a 50% probability in the later branching, with the other 50% for the medium value. This makes the medium branch have a 50% probability in total, while the two extreme values both have a 25% probability. For the climate sensitivity, the first branching is with a 50% probability for both branches. However, the second branching is different. If the high sensitivity is excluded in the first branch, there is a 21% conditional probability of the low branch in the second branching, meaning a 10.5% total probability for the low sensitivity. Similarly, there is a 23% conditional probability of high damages in the other branch, resulting in a 11.5% probability for the high sensitivity. The remaining 78% is the final probability of medium sensitivity.

Table B.3: Climate sensitivity and damage exponent values

	Climate sensitivity	Damage exponent
High	6	4
Medium	3	2
Low	1.5	1

This uncertain process is modeled by means of a multi-stage stochastic programming problem, where new information is obtained gradually. It is possible to perform research on these parameters. If the research succeeds, one of the extreme values is excluded already before 2050, revealing the first branching in the observation process earlier than without or with failed research. The observation of the actual parameter value (the second branching) still happens after 2070, after all abatement decisions have been made. The total cost we aim to minimize is then a discounted sum of research costs, abatement costs (B.3) for years 2030, 2050 and 2070, and damage costs (B.4).

References

Alonso-Ayuso, A., Escudero, L.F., Garìn, A., Ortuño, M.T., Pérez, G., 2003. An approach for strategic supply chain planning under uncertainty based on stochastic 0-1 programming. Journal of Global Optimization 26, 97–124. https://doi.org/10.1023/A:1023071216923.

Apap, R.M., Grossmann, I.E., 2017. Models and computational strategies for multistage stochastic programming under endogenous and exogenous uncertainties. Computers & Chemical Engineering 103, 233–274. https://doi.org/10.1016/j.compchemeng.2016.11.011.

Baker, E., Bosetti, V., Anadon, L.D., Henrion, M., Aleluia Reis, L., 2015. Future costs of key low-carbon energy technologies: Harmonization and aggregation of energy technology expert elicitation data. Energy Policy 80, 219–232. https://doi.org/10.1016/j.enpol.2014.10.008.

Bellman, R., 1961. Adaptive Control Processes: A Guided Tour. Princeton University Press.

Bellman, R., 1966. Dynamic programming. Science 153, 34–37. https://doi.org/10.1126/science.153.3731.34.

Benders, J.F., 1962. Partitioning procedures for solving mixed-variables programming problems. Numerische mathematik 4, 238–252.

Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B., 2017. Julia: A fresh approach to numerical computing. SIAM Review 59, 65–98. https://doi.org/10.1137/141000671.

Birge, J.R., Louveaux, F., 2011. Introduction to stochastic programming. Springer Science & Business Media.

Boland, N., Dumitrescu, I., Froyland, G., 2008. A multistage stochastic programming approach to open pit mine production scheduling with uncertain geology. Optimization online , 1–33.

Calvin, K., Cowie, A., Berndes, G., Arneth, A., Cherubini, F., Portugal-Pereira, J., Grassi, G., House, J., Johnson, F.X., Popp, A., Rounsevell, M., Slade, R., Smith, P., 2021. Bioenergy for climate change mitigation: scale and sustainability. GCB Bioenergy 13, 1346–1371. https://doi.org/10.1111/gcbb.12863.

Carøe, C.C., Schultz, R., 1999. Dual decomposition in stochastic integer programming. Operations Research Letters 24, 37–45. https://doi.org/10.1016/S0167-6377(98)00050-9.

Colvin, M., Maravelias, C.T., 2010. Modeling methods and a branch and cut algorithm for pharmaceutical clinical trial planning using stochastic programming. European Journal of Operational Research 203, 205–215. https://doi.org/10.1016/j.ejor.2009.07.022.

Dantzig, G.B., 1955. Linear Programming under Uncertainty. Management Science 1, 197–206. https://doi.org/10.1287/mnsc.1.3-4.197.

Dantzig, G.B., Wolfe, P., 1960. Decomposition principle for linear programs. Operations Research 8, 101–111. https://doi.org/10.1287/opre.8.1.101.

Dunning, I., Huchette, J., Lubin, M., 2017. JuMP: A Modeling Language for Mathematical Optimization. SIAM Review 59, 295–320. https://doi.org/10.1137/15M1020575.

Dupačová, J., 2006. Optimization under exogenous and endogenous uncertainty, in: Proceedings of the 24th International conference on Mathematical Methods in Economics, pp. 131–136. https://doi.org/10.13140/2.1.2682.2089.

Ekholm, T., 2018. Climatic Cost-benefit Analysis Under Uncertainty and Learning on Climate Sensitivity and Damages. Ecological Economics 154, 99–106. https://doi.org/10.1016/j.ecolecon.2018.07.024.

Ekholm, T., Baker, E., 2022. Multiple Beliefs, Dominance and Dynamic Consistency. Management Science 68, 529–540. https://doi.org/10.1287/mnsc.2020.3908.

Fuss, S., Lamb, W.F., Callaghan, M.W., Hilaire, J., Creutzig, F., Amann, T., Beringer, T., De Oliveira Garcia, W., Hartmann, J., Khanna, T., Luderer, G., Nemet, G.F., Rogelj, J., Smith, P., Vicente, J.V., Wilcox, J., Del Mar Zamora Dominguez, M., Minx, J.C., 2018. Negative emissions - Part 2: Costs, potentials and side effects. Environmental Research Letters 13. https://doi.org/10.1088/1748-9326/aabf9f.

Goel, V., Grossmann, I.E., 2006. A Class of stochastic programs with decision dependent uncertainty. Math. Program., Ser. B 108, 355–394. https://doi.org/10.1007/s10107-006-0715-7.

Gurobi Optimization, LLC, 2022. Gurobi Optimizer Reference Manual. URL: `https://www.gurobi.com`.

Gustafsson, J., Salo, A., 2005. Contingent portfolio programming for the management of risky projects. Operations research 53, 946–956. https://doi.org/10.1287/opre.1050.0225.

Hellemo, L., Barton, P.I., Tomasgard, A., 2018. Decision-dependent probabilities in stochastic programs with recourse. Computational Management Science 15, 369–395. https://doi.org/10.1007/s10287-018-0330-0.

Herrala, O., 2020. An efficient strategy for solving stochastic programming problems under endogenous and exogenous uncertainties. Master's thesis. Aalto University.

Herrala, O., 2022. Source code repository: Decision Programming extensions. URL: `https://github.com/solliolli/decision-programming-extensions`.

Howard, R.A., Matheson, J.E., 2005. Influence diagrams. Decision Analysis 2, 127–143. https://doi.org/10.1287/deca.1050.0020.

Huang, K., Ahmed, S., 2009. The value of multistage stochastic programming in capacity planning under uncertainty. Operations Research 57, 893–904. https://doi.org/10.1287/opre.1080.0623.

Jonsbråten, T.W., Wets, R.J., Woodruff, D.L., 1998. A class of stochastic programs with decision dependent random elements. Annals of Operations Research 82, 83–106. https://doi.org/10.1023/A:1018943626786.

Lauritzen, S.L., Nilsson, D., 2001. Representing and solving decision problems with limited information. Management Science 47, 1235–1251. https://doi.org/10.1287/mnsc.47.9.1235.9779.

Lee, S., Grossmann, I.E., 2000. New algorithms for nonlinear generalized disjunctive programming. Computers & Chemical Engineering 24, 2125–2141. https://doi.org/10.1016/S0098-1354(00)00581-0.

Lu, R., Ding, T., Qin, B., Ma, J., Fang, X., Dong, Z., 2019. Multi-stage stochastic programming to joint economic dispatch for energy and reserve with uncertain renewable energy. IEEE Transactions on Sustainable Energy 11, 1140–1151. https://doi.org/10.1109/TSTE.2019.2918269.

Nordhaus, W.D., 2017. Revisiting the social cost of carbon. Proceedings of the National Academy of Sciences 114, 1518–1523. https://doi.org/10.1073/pnas.1609244114.

Ntaimo, L., Arrubla, J.A.G., Stripling, C., Young, J., Spencer, T., 2012. A stochastic programming standard response model for wildfire initial attack planning. Canadian Journal of Forest Research 42, 987–1001. https://doi.org/10.1139/x2012-032.

Oliveira, F., Herrala, O., Tollander de Balsch, J., Hankimaa, H., Andelmin, J., 2021. DecisionProgramming.jl v0.1.0. Zenodo https://doi.org/10.5281/zenodo.6182560.

Peeta, S., Sibel Salman, F., Gunnec, D., Viswanath, K., 2010. Pre-disaster investment decisions for strengthening a highway network. Computers and Operations Research 37, 1708–1719. https://doi.org/10.1016/j.cor.2009.12.006.

Pereira, M.V., Pinto, L.M., 1991. Multi-stage stochastic optimization applied to energy planning. Mathematical programming 52, 359–375. https://doi.org/10.1007/BF01582895.

Powell, W.B., 2007. Approximate Dynamic Programming: Solving the curses of dimensionality. John Wiley & Sons.

Rogelj, J., McCollum, D.L., Reisinger, A., Meinshausen, M., Riahi, K., 2013. Probabilistic cost estimates for climate change mitigation. Nature 493, 79–83. https://doi.org/10.1038/nature11787.

Rogelj, J., Popp, A., Calvin, K.V., Luderer, G., Emmerling, J., Gernaat, D., Fujimori, S., Strefler, J., Hasegawa, T., Marangoni, G., Krey, V., Kriegler, E., Riahi, K., Van Vuuren, D.P., Doelman, J., Drouet, L., Edmonds, J., Fricko, O., Harmsen, M., Havlík, P., Humpenöder, F., Stehfest, E., Tavoni, M., 2018. Scenarios towards limiting global mean temperature increase below 1.5 C. Nature Climate Change 8, 325–332. https://doi.org/10.1038/s41558-018-0091-3.

Ruszczyński, A., 1997. Decomposition methods in stochastic programming. Mathematical Programming, Series B 79, 333–353. https://doi.org/10.1007/BF02614323.

Ruszczyński, A., Shapiro, A., 2003. Stochastic programming models. Handbooks in operations research and management science 10, 1–64. https://doi.org/10.1016/S0927-0507(03)10001-1.

Salo, A., Andelmin, J., Oliveira, F., 2022. Decision programming for mixed-integer multi-stage optimization under uncertainty. European Journal of Operational Research 299, 550–565. https://doi.org/10.1016/j.ejor.2021.12.013.

Shachter, R.D., 1986. Evaluating Influence Diagrams. Operations Research 34, 871–882. https://doi.org/10.1287/opre.34.6.871.

Solak, S., Clarke, J.P., Johnson, E., Barnes, E., 2008. A stochastic programming model with decision dependent uncertainty realizations for technology portfolio management, in: Operations Research Proceedings 2007, Springer. pp. 75–80. https://doi.org/10.1007/978-3-540-77903-2_12.

Tarhan, B., Grossmann, I.E., Goel, V., 2009. Stochastic programming approach for the planning of offshore oil or gas field infrastructure under decision-dependent uncertainty. Industrial and Engineering Chemistry Research 48, 3078–3097. https://doi.org/10.1021/ie8013549.

Walkup, D.W., Wets, R.J.B., 1967. Stochastic Programs with Recourse. SIAM Journal on Applied Mathematics 15, 1299–1314. https://doi.org/10.1137/0115113.

Wallace, S.W., Ziemba, W.T., 2005. Applications of stochastic programming. SIAM.

Yin, L., Han, L., 2020. International assets allocation with risk management via multi-stage stochastic programming. Computational Economics 55, 383–405. https://doi.org/10.1007/s10614-013-9365-z.