Math. Program., Ser. B (2016) 157:69–93
DOI 10.1007/s10107-015-0970-6

FULL LENGTH PAPER

Minimum cardinality non-anticipativity constraint sets for multistage stochastic programming

Natashia Boland[1,6] · **Irina Dumitrescu**[2,3] · **Gary Froyland**[4] · **Thomas Kalinowski**[5]

Received: 11 February 2014 / Accepted: 15 December 2015 / Published online: 7 January 2016
© Springer-Verlag Berlin Heidelberg and Mathematical Optimization Society 2016

Abstract We consider multistage stochastic programs, in which decisions can adapt over time, (i.e., at each stage), in response to observation of one or more random variables (uncertain parameters). The case that the time at which each observation occurs is decision-dependent, known as stochastic programming with endogeneous observation of uncertainty, presents particular challenges in handling non-anticipativity. Although such stochastic programs can be tackled by using binary variables to model the time at which each endogenous uncertain parameter is observed, the consequent *conditional* non-anticipativity constraints form a very large class, with cardinality in the order of

This research was supported by the Australian Research Council Linkage Project grant LP0561744 and by BHP Billiton Limited.

✉ Natashia Boland
natashia.boland@isye.gatech.edu

Irina Dumitrescu
irina.extra.mail@gmail.com

Gary Froyland
g.froyland@unsw.edu.au

Thomas Kalinowski
thomas.kalinowski@newcastle.edu.au

1 The University of Newcastle, University Drive, Callaghan, NSW 2308, Australia

2 IBM Research - Australia, Carlton, Australia

3 University of Melbourne, Melbourne, Australia

4 The University of New South Wales, Sydney, Australia

5 The University of Newcastle, Callaghan, Australia

6 H. Milton Stewart School of Industrial & Systems Engineering Georgia Institute of Technology, Atlanta, USA

the square of the number of scenarios. However, depending on the properties of the set of scenarios considered, only very few of these constraints may be required for validity of the model. Here we characterize minimal sufficient sets of non-anticipativity constraints, and prove that their matroid structure enables sets of minimum cardinality to be found efficiently, under general conditions on the structure of the scenario set.

Keywords Stochastic programming · Endogeneous uncertainty · Multistage stochastic programming

1 Introduction

Stochastic programming provides an approach to decision-making that takes account of the probability distributions of uncertain parameters. Typically the values of these parameters are revealed over time, or with stage in a multistage decision setting, and decisions made at each stage hedge against possible realizations of parameters revealed in future stages. Of enormous practical importance, such multistage stochastic programming models have been studied for many years now, with the literature offering a wealth of theoretical and algorithmic tools for solving them: see, for example, [4,25,26,28,30,32] and the many references therein. Naturally, given the challenging nature of these problems, decomposition approaches have been a primary focus of research in multistage stochastic programming, with a recent surge of progress [20,29,31], especially in the direction of extensions to risk averse measures [2,16,22,23,31,33], and with increasingly vigorous interest in decision rules [3,11].

The majority of this literature assumes that the sources of uncertainty are exogeneous to the decision-making process; relatively little attention has, to date, been paid to stochastic programming in the presence of *endogenous uncertainty*. Endogeneous uncertainty is defined to occur when the underlying stochastic process depends on the optimization decisions [21]. This can occur via direct alteration of the probability distributions, or by determination of the timing of parameter uncertainty resolution (referred to as *type 1* and *type 2* forms respectively [14]). The latter form is known as *endogeneous observation of uncertainty*. In such cases, one might see a further influence of stochasticity on decision-making: as noted, for example, in Artstein and Wets [1], one could, and perhaps should, invest some resources in estimation of parameters defining the uncertainty.

The importance of stochastic programming with endogeneous uncertainty is underscored by the rapidly increasing body of literature on its applications. It has been used for gas field development planning by Goel and Grossmann [13] and Goel et al. [15], for the planning of clinical trials in the pharmaceutical R&D pipeline by Colvin and Maravelias [7–9], for scheduling the production of open pit mines by Boland et al. [5,24], for process network planning by Tarhan and Grossmann [35] and Gupta and Grossmann [18], for oil and gas field investment and operation by Tarhan et al. [36], Vayanos et al. [38] and Gupta [17], for design of groundwater well systems by Giles [12], for R&D technology project portfolio management by Solak et al. [34], for workforce capacity planning by Fragnière et al. [10], and for scheduling hospital operating theatres by Bruni et al. [6]. As an illustration, Fragnière et al. [10] show that

if the dependence of future demand for the services of an enterprise on the quality of the service it provides, (a function of the workforce expertise decided in the model), is ignored, so as to remove the endogeneous uncertainty, the number of qualified workers needed is underestimated by the model by more than 30 %.

As noted, for example by Jonsbråten et al. [21], endogenous uncertainty makes the stochastic program substantially more difficult to solve, and even relatively recent papers (e.g. [19]) comment on the sparsity of literature addressing such cases. A key challenge in the solution of such problems is the large number of non-anticipativity constraints (NACs). In the case of endogeneous uncertainty, additional binary variables and logical constraints are required to model non-anticipativity, with the deterministic equivalent problem size (number of variables and constraints) increasing with the number of scenarios. Thus solution approaches have focussed on ways of addressing the challenge of NACs.

Lagrangian relaxation of the NACs to yield a decomposition by scenario has proved particularly useful. In [14,36,37] it is applied in conjunction with a duality based search to close the gap. It is also used in [34], which applies sample average approximation with Lagrangian relaxation to solve the sample problems and branch and bound to close the gap. Most recently, the concept has been generalized by Gupta and Grossmann [19], who exploit the structure of their scenario space to permit Lagrangian decomposition into scenario group subproblems, with the NACs explicitly retained within each group but Lagrangian relaxed between the groups.

Delaying the addition of NACs until needed has been another strategy for dealing with their challenge. In [9] a branch and cut algorithm first removes necessary non-anticipativity constraints that are unlikely to be active from the initial formulation and then only adds them if they are violated within the search tree, with different approaches to constraint violation checking explored to improve performance. Gupta and Grossmann [18] consider (in addition to Lagrangian relaxation) a strategy where the problem is solved with the NACs included only for the stages up to a specified number (k), with k incremented and the process repeated until the solution has no realization of an uncertain parameter occurring after stage k. A two-phase NAC relaxation strategy is also proposed, with NACs first treated with a cutting plane approach (repeatedly solving the LP relaxation, adding violated NACs until all are satisfied), and then added to the MIPs if needed. All three approaches are very effective in reducing solve times on two process network examples.

Most approaches to date focus on exact solution for uncertain parameters having discrete distributions: a novel approach is taken by Vayanos et al. [38], who show how to approximate the problem via decision rules, using piecewise constant and piecewise linear functions of the uncertainties. Their techniques extend to continuously distributed uncertain parameters via partition of the uncertainty domain.

All these approaches will benefit when the set of NACs that needs to be considered is reduced, for example by observing that some NACs are implied by others, and so are not necessary in the formulation. Key steps in this direction were developed in [5,9,14,18]. All observed that substantial proportions of the NACs could safely be ignored, but used the structure of the application ([5,9]) or the structure of the scenario space ([9,14,18]) to determine this. For example, the properties given in [14,18] all depend on the scenario space being the cross product of realizations of all uncertain parameters.

Work to date in this direction has largely demonstrated substantial computational benefits from reduction in the number of conditional NACs included. Goel and Grossmann [14] report on a capacity expansion problem in process networks, in which installation/expansion of three processing units is to be planned over 10 periods, finding that a reduction in the number of NACs of 32.5 % reduced the MIP solver computing time by 44.4 %. In open-pit mining problems, even with a very small number of scenarios, a speed-up of more than 30 % in MIP solver time as a consequence of halving the number of NACs is observed [5]. The positive impact of reduction in NACs becomes even more apparent as the size of the scenario space increases. The work of Colvin and Maravelius [7,9], in the context of pharmaceutical drug testing, is particularly striking in this regard. In [7], instances, involving 3 drugs over a 12-period planning horizon and 5 drugs over a 6-period horizon, generated 37 million and 155 billion NACs, respectively, so reduction of these was essential. More comprehensive tests, in [9], compared three levels of reduction, and showed that at each level, computation times decreased, and, more importantly, larger problems could be loaded into memory, and so could be solved. Even after three levels of reduction, the hardest instance still had 1.3 million NACs, and hence a specialized branch-and-bound procedure was developed.

Our primary contribution is to provide general conditions under which NACs can safely be omitted from the formulation: we define *generator sets* of NACs, and show that these are sufficient to imply all NACs. The generator set property is also necessary: any subset of NACs that guarantees all are satisfied must be a generator set. The generator set property does not rely on any structure on the scenario space, however in the case of the cross product scenario space, we recover the properties of [18] as a special case, and generalize these. We show further that generator sets have a matroid property, and so minimum cardinality generator sets can be found efficiently by a greedy algorithm. The generality of our results could be particularly helpful in the context of recent algorithmic ideas: both sample average approximation (e.g. [34]) and scenario group Lagrangian decomposition ([19]) solve problems over subsets of scenarios; our results give an efficient method of identifying a minimum cardinality set of NACs to include in a problem over any scenario subset.

2 A generic model

In general multistage stochastic programming with endogeneous observation of uncertainty, there is a finite set of sources of uncertainty, indexed by $\mathcal{I}$, and, for each $i \in \mathcal{I}$, a random variable $\tilde{\theta}_i$. The set of possible realizations for $\tilde{\theta}_i$ is given by Θ_i. A scenario $s \in \mathcal{S}$ is characterized by a vector of realizations $(\theta_i^s)_{i\in\mathcal{I}}$, with $\theta_i^s \in \Theta_i$ for each $i \in \mathcal{I}$. A *differentiator set* $\mathcal{D}(r, s)$ is the set of all $i \in \mathcal{I}$ such that observation of i allows scenarios s and r to be differentiated. We assume that $\mathcal{D}(r, s) \neq \emptyset$ for all distinct scenario pairs $\{r, s\}$. Usually this is simply the set of i for which θ_i^s and θ_i^r are different, however in some cases, such as that of uncertain geology in open-pit mining [5], some notion of "sufficiently different", such as $|\theta_i^r - \theta_i^s| > \epsilon$ for suitable $\epsilon > 0$, may be of interest. (The ideas of [38] are also relevant here.) We assume that differentiator sets are symmetric, i.e. satisfy $\mathcal{D}(r, s) = \mathcal{D}(s, r)$ for any $r, s \in \mathcal{S}, r \neq s$.

To illustrate these concepts and our notation, and to highlight the differences between endogeneous observation of uncertainty in multistage stochastic programs, and the more well-studied, exogeneous, form of uncertainty, we introduce an example application of the former: the game of Hangman. In Hangman, the player tries to guess a word selected by an opponent. Initially, the player knows only the number of letters in the word. The player then guesses a letter of the alphabet, and the opponent must reveal to the player the positions in the word, if any, in which that letter appears. Usually the player is allowed a maximimum number of incorrect guesses, (guesses of letters that do not appear in the word), and loses if he cannot guess the word before these are exhausted. Here we use a simpler variant: the player is allowed a fixed number of guesses, irrespective of whether they are correct or not. This game can be modeled as a multistage stochastic program with endogeneous observation of uncertainty, as follows. Take the scenarios to be the words available to the opponent, so $\mathcal{S}$ can be thought of as a dictionary, consisting of a set of words, each of the given length, L letters, over an alphabet, $\mathcal{I}$, of possible letters. For each letter, $i \in \mathcal{I}$, there is a set of possible patterns for that letter: $\Theta_i \subseteq \{0,1\}^L$. At each stage of the game, $t = 1, \ldots, T$, the player must choose a letter $i \in \mathcal{I}$, resulting in the revelation of its realization. The player may adapt his guesses based (only) on what he has learned about the word so far. A specific example is given below.

Example Game of Hangman Suppose the opponent uses the dictionary $\mathcal{S} = \{\texttt{neat, nest, sate, seat, sent, teat, tent, test}\}$, of eight four-letter words, over the alphabet $\mathcal{I} = \{\texttt{a,e,n,s,t}\}$. Then the possible realizations of letter patterns are $\Theta_{\texttt{a}} = \{(0,0,0,0), (0,1,0,0), (0,0,1,0)\}$, $\Theta_{\texttt{e}} = \{(0,1,0,0), (0,0,0,1)\}$, $\Theta_{\texttt{t}} = \{(1,0,0,0), (0,0,1,0), (0,0,0,1), (1,0,0,1)\}$, etc. A scenario corresponds to a realization for each of the random variables, so, for example, $\theta^{\texttt{tent}}$ is given by $\theta^{\texttt{tent}}_{\texttt{t}} = (1,0,0,1)$, $\theta^{\texttt{tent}}_{\texttt{e}} = (0,1,0,0)$, $\theta^{\texttt{tent}}_{\texttt{n}} = (0,0,1,0)$, and $\theta^{\texttt{tent}}_{i} = (0,0,0,0)$ for $i \in \{\texttt{a,s}\}$. The letters in the differentiator sets for each pair of words (scenarios) are shown in Table 1. For example, `nest` and `sent` can only be differentiated by guessing either `n` or `s`, since $\theta^{\texttt{nest}}_{\texttt{n}} \neq \theta^{\texttt{sent}}_{\texttt{n}}$ and $\theta^{\texttt{nest}}_{\texttt{s}} \neq \theta^{\texttt{sent}}_{\texttt{s}}$, while $\theta^{\texttt{nest}}_{i} = \theta^{\texttt{sent}}_{i}$ for all $i \in \mathcal{I}\backslash\{\texttt{n,s}\}$, so $\mathcal{D}(\texttt{nest,sent}) = \{\texttt{n,s}\}$.

Suppose the player is allowed 3 guesses, so $T = 3$. If the player guesses `n` first, then, depending on the scenario, (word chosen by the opponent), the player will learn, after the first stage, that the scenario is either one of {`neat,nest`}, one of {`sent,tent`},

Table 1 Letters in the differentiator sets for each pair of words in the Hangman example dictionary

	nest	sate	seat	sent	teat	tent	test
neat	a,s	a,e,n,s,t	n,s	a,n,s	n,t	a,n,t	a,n,s,t
nest	–	a,e,n,s,t	a,n,s	n,s	a,n,s,t	n,s,t	n,t
sate		–	a,e,t	a,e,n,t	a,e,s,t	a,e,n,s,t	a,e,s,t
seat			–	a,n	s,t	a,n,s,t	a,s,t
sent				–	a,n,s,t	s,t	n,s,t
teat					–	a,n	a,s
tent						–	n,s

Symmetric entries are omitted

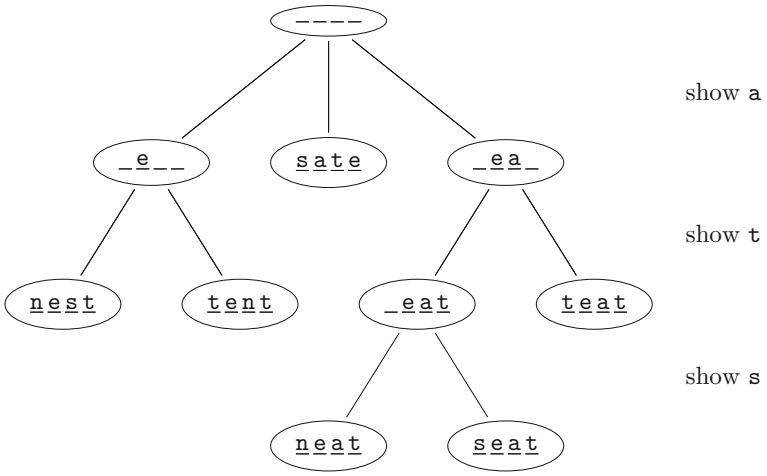

Fig. 1 A scenario tree for the Hangman example, for letters revealed in the fixed sequence (a, t, s, n, e). Subtrees that are paths have been contracted to their root node. All letters are shown as soon as their choice is forced

or is one of {sate,seat,teat,test}. In the first case, guessing t next is pointless, but in the latter two cases, guessing t in stage 2 will be helpful: it will reveal whether the word is sent or tent in the second case, or {sate}, {seat}, or one of {teat,test} in the third case.

This example highlights an important difference from the case of exogeneous uncertainty. In exogeneous uncertainty, scenarios are often modeled using a scenario tree. Each level of the tree corresponds to both a decision stage, and a stage at which uncertainty is (realized and) observed. In the case of endogeneous observation of uncertainty, the stage at which an uncertain parameter is observed cannot be known in advance; it depends on the decisions made. Thus, even if the scenarios can be represented using a tree, there may be no connection between the levels of the tree, and when sources of uncertainty are observed. In the Hangman example, there is no a priori reason to construct the scenarios using a scenario tree. One could be constructed a posteriori by fixing a sequence in which the letters are revealed. Such a tree is shown in Fig. 1. However, the first stage decision of the player may be, for example, the third letter in the scenario tree sequence: the sequence in which the letters are revealed during the game is precisely the decision problem.

This "divorce" of the scenarios from the decision stages is typical in an endogeneous uncertainty setting. For example, in open-pit mining, what is observed of a realization of the underground geology depends on where the decision has been made to dig, and on how long that takes. In pharmaceutical drug testing, [9], what is learned at each stage about which drugs will succeed in a trial depends on which drugs were selected for testing, and on when their preceding trials were scheduled. In general, we might say that what the decision-maker learns about the scenario depends on where her decisions cause her to look.

Since non-anticipativity constraints are typically encoded, in the exogeneous case, in the scenario tree, this separation of scenarios from decision stages illustrates why a

quite different approach to enforcing non-anticipativity is needed in the endogeneous case.

Before discussing how this may be done in general, we explain the concept using the Hangman game, by formulating a deterministic equivalent model for the problem. In what follows, and throughout this paper, we use the notation $[n]$, for n a positive integer, to denote the set $\{1, \ldots, n\}$. First, to model the decisions at each stage, we introduce binary variables, $\phi^s_{i,t}$, for each $s \in \mathcal{S}$, $i \in \mathcal{I}$, and $t \in [T]$, set to 1 if the player guesses letter i in his tth guess, under scenario s, and zero otherwise. Since the player cannot choose more than one letter per guess, we require

$$\sum_{i \in \mathcal{I}} \phi^s_{i,t} \leq 1, \quad \forall t \in [T], \quad \forall s \in \mathcal{S}.$$

Other constraints may be desired, and other variables, depending on how an objective is modeled; we omit these, as they are not helpful for explaining the key concepts. Instead we focus on non-anticipativity: we must model the requirement that the guess made by the player in stage t under scenarios r and s must be the same if no letter has been guessed so far that distinguishes between the two scenarios. In other words, for each pair of distinct scenarios $r, s \in \mathcal{S}$ and each stage $t \in [T]$, we require that

$$\begin{aligned} &\text{if } \phi^s_{i,t'} = \phi^r_{i,t'} = 0 \quad \text{for all } i \in \mathcal{D}(r,s) \text{ and all } t' \in [t-1], \\ &\text{then } \phi^s_{i,t} = \phi^r_{i,t} \quad \text{for all } i \in \mathcal{I}. \end{aligned}$$

To model this, it is convenient to introduce binary variables, $\beta^s_{i,t}$, for each letter, $i \in \mathcal{I}$, scenario, $s \in \mathcal{S}$, and stage, $t \in [T]$, set to 1 if the player guesses i in any stage up to t, and zero otherwise:

$$\beta^s_{i,t} = \sum_{t'=1}^{t-1} \phi^s_{i,t'}, \qquad \forall t \in [T], \quad \forall s \in \mathcal{S}, \quad \forall i \in \mathcal{I}. \tag{1}$$

(Since $\beta^s_{i,t}$ is binary, this implicitly requires the right-hand side to be at most one, which enforces the player to guess a letter in at most one stage; this is expected anyway.) The condition that $\phi^s_{i,t'} = \phi^r_{i,t'} = 0$ for all $i \in \mathcal{D}(r,s)$ and $t' \in [t-1]$ is then equivalent to $\sum_{i \in \mathcal{D}(r,s)} \beta^\sigma_{i,t} = 0$, for $\sigma = r$ and $\sigma = s$. In fact, either choice of σ suffices; one cannot hold without the other. This is discussed further, later. Now non-anticipativity asks that, for each pair of distinct scenarios $r, s \in \mathcal{S}$ and each stage $t \in [T]$,

$$\text{if } \sum_{i \in \mathcal{D}(r,s)} \beta^s_{i,t} = 0 \text{ then } \phi^s_{i,t} = \phi^r_{i,t} \text{ for all } i \in \mathcal{I}.$$

In general, in the endogeneous case, since the stage of the stochastic program at which $\tilde{\theta}_i$ is observed, depends on decisions made by the model, it is natural to model the uncertainty via a binary variable $\beta^s_{i,t}$ for each scenario $s \in \mathcal{S}$, each source of uncertainty $i \in \mathcal{I}$ and each stage $t \in [T]$, to indicate whether or not $\tilde{\theta}_i$ has been observed by stage t under scenario s. We refer to these as *observation indicator*

variables. In some applications, for example in open-pit mine production scheduling [5] and pharmaceutical testing [9], such variables naturally occur in the model; in others, they may need to be added. (In the Hangman model, they can be substituted out using Eq. (1)). If a source of uncertainty has been observed by some stage, then it has certainly been observed by all later stages, thus the observation indicator variables must satisfy

$$\beta^s_{i,t} \le \beta^s_{i,t+1}, \qquad \forall i \in \mathcal{I}, \forall s \in \mathcal{S}, \forall t \in [T-1]. \tag{2}$$

Let ϕ^s_t for each $s \in \mathcal{S}$ and $t \in [T]$ be the vector of all scenario-dependent variables in the model that cannot be different from those in another scenario unless the scenarios have been differentiated by stage t. As in the Hangman example, the condition $\sum_{i\in\mathcal{D}(r,s)} \beta^s_{i,t} = 0$ is equivalent to the condition that scenarios r and s have not been differentiated by stage t, so the non-anticipativity constraints must be applied when this condition is satisfied. Then the general logical form of conditional non-anticipativity constraints is

$$\sum_{i\in\mathcal{D}(r,s)} \beta^s_{i,t} = 0 \;\Rightarrow\; \phi^s_t = \phi^r_t, \qquad \forall \{r,s\} \in \mathcal{T}, \forall t \in [T], \tag{3}$$

and the natural model of this requirement is the usual linearization of

$$|\phi^s_{l,t} - \phi^r_{l,t}| \le M^{\{r,s\}}_{l,t} \sum_{j\in\mathcal{D}(r,s)} \beta^s_{j,t}, \qquad \forall l \in \mathcal{L}_t, \{r,s\} \in \mathcal{T}, t \in [T], \tag{4}$$

where $\mathcal{L}_t$ is the index set for the ϕ^s_t vector and $\mathcal{T} = \{\{r,s\} \subseteq \mathcal{S} : r \ne s\}$ denotes the set of all scenario pairs. This is a "big-M" constraint, and $M^{\{r,s\}}_{l,t}$ can be taken to be any known upper bound on the value of $|\phi^s_{l,t} - \phi^r_{l,t}|$.

For simplicity of exposition, we omit discussion of decision variables that do not depend on the observation of uncertainty; these can be included in a straightforward way.

Non-anticipativity is also required for the observation indicator variables. If at some stage t, two scenarios have not yet been distinguished, then at the next stage, $t+1$, the information observed must be the same under each of these two scenarios; either a source of uncertainty has been observed under both scenarios, or it has not. Here we are now more careful about the scenario superscript used in the condition under which non-anticipativity is required: we ask here that both $\sum_{i\in\mathcal{D}(r,s)} \beta^s_{i,t} = 0$ and $\sum_{i\in\mathcal{D}(r,s)} \beta^r_{i,t} = 0$ hold, to indicate that r and s have not yet been differentiated, and then discuss how only one suffices. The logical form of the non-anticipativity constraints for the observation indicator variables is then

$$\sum_{j\in\mathcal{D}(r,s)} \beta^\sigma_{j,t} = 0 \;\Rightarrow\; \beta^s_{i,t+1} = \beta^r_{i,t+1}, \forall i \in \mathcal{I}, \forall \{r,s\} \in \mathcal{T}, \forall \sigma \in \{r,s\}, \forall t \in [T-1], \tag{5}$$

and the natural model of this requirement is the usual linearization of

$$|\beta^s_{i,t+1} - \beta^r_{i,t+1}| \leq \sum_{j \in \mathcal{D}(r,s)} \beta^\sigma_{j,t}, \ \forall i \in \mathcal{I}, \{r,s\} \in \mathcal{T}, \forall \sigma \in \{r,s\}, \forall t \in [T-1], \quad (6)$$

together with the initialization

$$\beta^s_{i,1} = 0, \qquad \forall i \in \mathcal{I}, \forall s \in \mathcal{S}. \quad (7)$$

In fact, only *one* non-anticipativity constraint from (6) needs to be included in the model for each unordered scenario pair in $\mathcal{T}$, rather than the two given, and the choice of which to include may be arbitrary. The following result is straightforward to prove by induction; for completeness we provide the proof in the appendix.

Proposition 1 *Let β satisfy (2) and (7), and suppose that for every unordered scenario pair $\{r, s\} \in \mathcal{T}$,*

$$\sum_{j \in \mathcal{D}(r,s)} \beta^\sigma_{j,t} = 0 \ \Rightarrow \ \beta^s_{i,t+1} = \beta^r_{i,t+1}, \ \forall i \in \mathcal{I}, \forall t \in [T-1] \quad (8)$$

is satisfied for $\sigma = r$ or $\sigma = s$. Then (6) holds.

This result also shows that (4) is well-defined, in the sense that the choice of superscript in the observation indicator variable on the right-hand side can be made arbitrarily. This result is equivalent to Property 1 of Gupta and Grossmann [18]; see also [14].

3 Necessary and sufficient sets of non-anticipativity constraints

The previous section shows that—at least in principle—endogeneous observation of uncertainty can be modeled linearly with the use of binary variables. However the model has a very large number of constraints: Eq. (4) contains $\Omega(|\mathcal{S}|^2 \sum_{t=1}^{T} |\mathcal{L}_t|)$ constraints. The primary difficulty is that the number of non-anticipativity constraints depends on the number of scenarios *squared*. In what follows here, we show that the structure of the scenarios, and the resulting relationships between the differentiator sets, can be used to substantially reduce the number of non-anticipativity constraints that must be included for a valid model; we characterize sets of non-anticipativity constraints (NACs) that are both necessary and sufficient to ensuring all NACs are satisfied.

It is helpful to consider the complete (undirected) graph $(\mathcal{S}, \mathcal{T})$ on the vertex set given by the scenarios. Differentiator sets are given by a function from the graph edges to the power set of $\mathcal{I}$, i.e. $\mathcal{D} : \mathcal{T} \to 2^{\mathcal{I}}$. This graph for the Hangman example is given in Fig. 2. Paths in the graph satisfying a differentiator set containment property will be particularly useful. We refer to path $P \subseteq \mathcal{T}$ in the graph between vertices r and s, i.e. having r and s as its two end nodes, as an *r-s-path*.

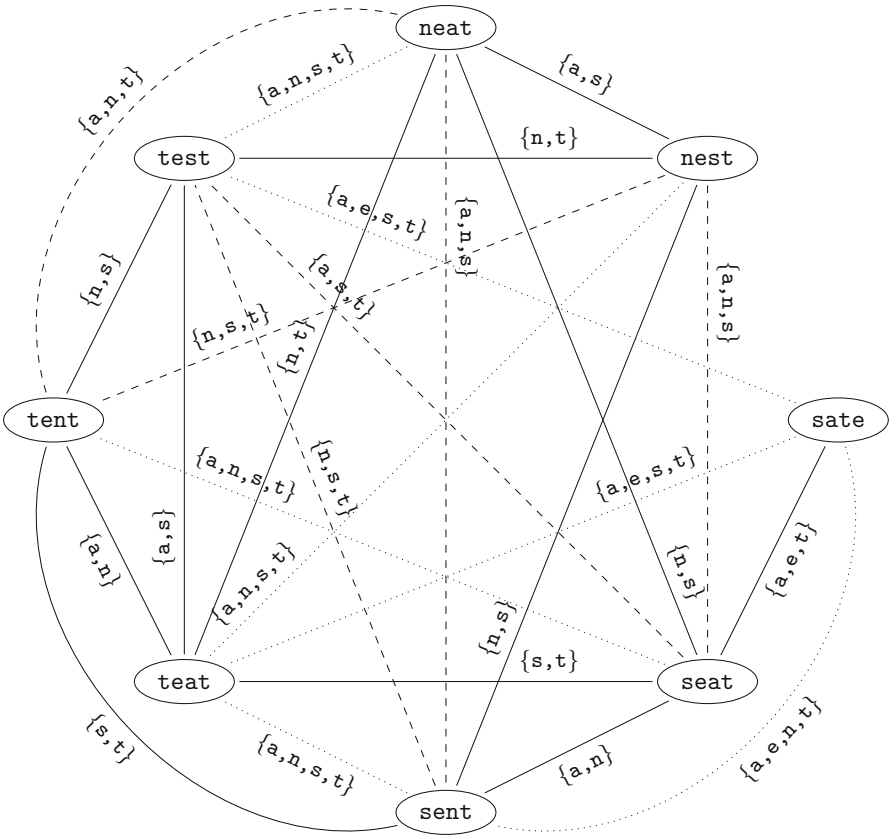

Fig. 2 The graph $(\mathcal{S}, \mathcal{T})$ for the Hangman example. The edge label for edge $\{r, s\}$ is the differentiator sets $\mathcal{D}(r, s)$ for the pair of scenarios r and s. *Dashed edges* indicate scenario pairs with differentiator set of cardinality 3, (except for {seat, sate}, which is *solid*), *dotted edges* indicate pairs with differentiator set of cardinality 4, and missing edges represent pairs for which the differentiator set is the entire set of 5 letters, $\mathcal{I}$. *Solid edges* show a minimum cardinality generator, which is, in this case, unique

3.1 Generator sets are necessary and sufficient

Since the key proof in this section relies only on the combinatorial structure of scenarios, and not on the formulation of the model of non-anticipativity, we present these results in terms of a general graph, $G = (V, E)$, and function, $\mathcal{D} : E \to 2^{[\mu]}$, from the edge set to the power set of the set $[\mu] = \{1, 2, \ldots, \mu\}$. However, these are to be applied to the node set $V = \mathcal{S}$, with $[\mu]$ identified with $\mathcal{I}$ and $\mathcal{D}$ defining the scenario differentiator sets identified by index.

The following definitions are with respect to $\mathcal{D}$, which is omitted in the notation because we fix one $\mathcal{D}$ throughout.

Definition 1 A pair (f, g) of functions $f : V \to \{0, 1\}^{\mu}$ and $g : V \to \mathbb{R}$ is called *E-consistent* if for every edge $e = \{v, w\} \in E$ we have the implication

$$\sum_{i \in \mathcal{D}(e)} f_i(v) = 0 \implies g(v) = g(w).$$

The NACs say that for the complete graph $(\mathcal{S}, \mathcal{T})$ on the scenario set, the pairs $(\beta_t, \beta_{i,t+1})$ for $t \in [T-1]$, $i \in \mathcal{I}$, and the pairs $(\beta_t, \phi_{l,t})$ for $t \in [T]$ and $l \in \mathcal{L}_t$ must be $\mathcal{T}$-consistent.

If the pair (f, g) is E-consistent then it is also E'-consistent for every subset $E' \subseteq E$. We call E' NAC-sufficient for G if the converse is also true. More precisely, we have the following definition.

Definition 2 An edge set $E' \subseteq E$ is *NAC-sufficient* for G if every E'-consistent pair (f, g) is also E-consistent.

Definition 3 Given a graph edge $e = \{v, w\} \in \mathcal{T}$, any v-w-path P having $\mathcal{D}(e') \subseteq \mathcal{D}(e)$ for all $e' \in P$ is called a *differentiator set containment path for* e, or *dsc-path for* e for short.

For example, in the Hangman instance, the path `test,sent,seat,sate` is not a dsc-path for {`test,sate`} since

$$\mathcal{D}(\texttt{sent,seat}) = \{\texttt{a,n}\} \nsubseteq \{\texttt{a,e,s,t}\} = \mathcal{D}(\texttt{test,sate}).$$

However, the path `test,teat,seat,sate` is a dsc-path for {`test,sate`}, since the differentiator sets for the edges on the path, {`a,s`}, {`s,t`} and {`a,e,t`}, are all subsets of $\mathcal{D}(\texttt{test,sate}) = \{\texttt{a,e,s,t}\}$.

Definition 4 An edge set $E' \subseteq E$ is a *generator* for $G = (V, E)$ if for every edge $e = \{v, w\} \in E$, there is a v-w-path P with $P \subseteq E'$ that is also a dsc-path for e.

A (minimal) generator for the Hangman example is indicated by the solid lines in Fig. 2. To check that this is a generator, simply confirm that, for each dashed, dotted or missing, edge, there is a dsc-path between its nodes consisting only of solid edges. Recall that a missing edge has differentiator set the entire set of letters, so a dsc-path for a missing edge is simply any path connecting its end points (the differentiator set containment property is automatically satisfied by any edge in the path).

Theorem 1 *An edge set $E' \subseteq E$ is NAC-sufficient for G if and only if it is a generator for G.*

Proof First let E' be a generator, and suppose the pair (f, g) is E'-consistent. Let $e = \{v, w\} \in E$ with $\sum_{i \in \mathcal{D}(e)} f_i(v) = 0$. Since E' is a generator there is a path $(v = v_0, v_1, \ldots, v_m = w)$ with $e_j = \{v_{j-1}, v_j\} \in E'$ and $\mathcal{D}(e_j) \subseteq \mathcal{D}(e)$ for $j \in \{1, \ldots, m\}$. Therefore, we have $\sum_{i \in \mathcal{D}(e_j)} f_i(v) = 0$ for all $j \in \{1, \ldots, m\}$. Since (f, g) is E'-consistent, this implies

$$g(w) = g(v_m) = g(v_{m-1}) = g(v_{m-2}) = \cdots = g(v_0) = g(v).$$

Thus the pair (f, g) is E-consistent, and E' is NAC-sufficient for G.

Conversely, let E' be NAC-sufficient for G, and assume that it is not a generator. Then there is some edge $e = \{v, w\} \in E \backslash E'$ such that there is no dsc-path for e in E'. Let V_0 be the set of all vertices $u \in V$ for which there exists a u-w-path P in E' such that $\mathcal{D}(e') \subseteq \mathcal{D}(e)$ for all $e' \in P$, including w itself. By assumption $v \notin V_0$. We define two functions $f : V \to \{0, 1\}^\mu$ and $g : V \to \{0, 1\}$ by

$$f_i(u) = \begin{cases} 0 & \text{for } i \in \mathcal{D}(e), \\ 1 & \text{otherwise,} \end{cases} \qquad u \in V,\ i \in [\mu], \quad \text{and}$$

$$g(u) = \begin{cases} 0 & \text{for } u \in V_0, \\ 1 & \text{for } u \in V \backslash V_0, \end{cases} \qquad u \in V.$$

Suppose the pair (f, g) is *not* E'-consistent. Then there must be a node $u \in V_0$ and a node $u' \in V \backslash V_0$ with $\{u, u'\} \in E'$ and $\sum_{i \in \mathcal{D}(u,u')} f_i(u) = 0$. But the latter implies $f_i(u) = 0$ for all $i \in \mathcal{D}(u, u')$, which is only possible if $i \in \mathcal{D}(u, u') \subseteq \mathcal{D}(e)$. Then since $u \in V_0$, by the definition of V_0, u' must be in V_0 too, giving a contradiction. Thus the pair (f, g) is E'-consistent.

However (f, g) is not E-consistent, since $\sum_{i \in \mathcal{D}(e)} f_i(v) = 0$ but $g(v) = 1 \neq 0 = g(w)$, which contradicts the assumption that E' is NAC-sufficient for G. □

A consequence of this theorem is that if $\mathcal{G}$ is a generator set for $(\mathcal{S}, \mathcal{T})$, it suffices to require NACs (3) and (5) only for scenario pairs in $\mathcal{G}$; those for pairs in $\mathcal{T} \backslash \mathcal{G}$ are implied by the constraints for pairs in $\mathcal{G}$. It also shows that any set of scenario pairs with the property that their NACs imply the rest must be a generator. In other words, it shows that applying NACs only for a subset of scenario pairs having the generator set property is necessary and sufficient for ensuring all NACs hold.

3.2 Stage-dependent generator sets and exogeneous uncertainty

So far, we have focused on subsets $\mathcal{G} \subseteq \mathcal{T}$ so that applying the NAC constraints for all pairs $\{r, s\} \in \mathcal{G}$ and all stages, t, ensures all NACs hold. However, a more general approach would be to seek subsets $\mathcal{G}^+ \subseteq \mathcal{T} \times [T]$ so that applying NACs only for $(\{r, s\}, t) \in \mathcal{G}^+$ suffices. In the case of endogeneous uncertainty discussed so far, it is not hard to see that nothing is lost by taking $\mathcal{G}^+ = \mathcal{G} \times [T]$ for $\mathcal{G}$ a generator of $\mathcal{T}$, since $\mathcal{G}^+ \subseteq \mathcal{T} \times [T]$ is sufficient if and only if $\mathcal{G}^+$ has the form

$$\mathcal{G}^+ = \bigcup_{t=1}^{T} (\mathcal{G}_t^+ \times \{t\})$$

where, for each $t \in [T]$, $\mathcal{G}_t^+$ is a generator for $\mathcal{T}$, Thus we may take $\mathcal{G}_t^+ = \mathcal{G}$ for any $\mathcal{G}$ a generator of $\mathcal{T}$, giving $\mathcal{G}^+ = \mathcal{G} \times [T]$.

In the case of exogeneous uncertainty, the situation is subtly different. Before discussing this further, we note that finding minimal ways of expressing NACs under exogeneous uncertainty is far less important than in the endogeneous case, since in

the latter, big-M constraints over binary variables are needed to model each NAC, whereas in the former, the NACs can be used to substitute out variables, leading to a more compact model. Nevertheless, it is instructive to consider how the concepts presented here apply in this, more widely studied, setting.

In the standard multistage stochastic programming setting, the random variables are indexed by the time periods, i.e., $\mathcal{I} = [T]$. In the first stage, non-anticipativity usually requires that

$$\phi_1^r = \phi_1^s, \qquad \forall \{r, s\} \in \mathcal{T}.$$

Here non-anticipativity is expressed using a constraint for every pair of scenarios. This corresponds to applying the constraint for every edge in the clique (complete graph) $(\mathcal{S}, \mathcal{T})$. But it is obvious that far fewer constraints suffice. For example, if the scenarios are ordered, with $\mathcal{S} = \{s_1, \ldots, s_{|\mathcal{S}|}\}$, then it suffices to require that

$$\phi_1^{s_k} = \phi_1^{s_{k+1}}, \qquad \forall k \in [|\mathcal{S}| - 1].$$

This corresponds to applying the NAC for every edge in a Hamiltonian path in $(\mathcal{S}, \mathcal{T})$. However, it is intuitively clear that applying the NAC for every edge in any spanning subgraph of $(\mathcal{S}, \mathcal{T})$ suffices, and this is what the generator set devolves to when non-anticipativity is unconditional. The differentiator set containment property of a dsc-path captures the requirement that if the condition under which an NAC for a scenario pair should be applied is met, then it is also met on all edges of the path connecting the two scenarios. Without the need for a condition to trigger application of the NAC, this part of the dsc-path definition can be dropped, and the only requirement for a generator is that there is a path between every pair of nodes (scenarios).

In general, for the exogeneous case, non-anticipativity can be captured as follows. We define a sequence of graphs G_1, G_2, …, G_T on the vertex set $\mathcal{S}$, where G_t has edge set

$$\mathcal{T}_t = \{\{r, s\} \ : \ \theta_i^r = \theta_i^s \text{ for } i = 1, \ldots, t\}.$$

In other words, each graph G_t is a node-disjoint union of cliques, one for each scenario tree node at stage t, and the cliques of G_t correspond to the scenario prefixes $(\theta_1, \ldots, \theta_t)$. The NACs have the form

$$\phi_t^r = \phi_t^s \quad \text{for all } r, s \in \mathcal{S} \text{ with } \theta_i^r = \theta_i^s \text{ for } i \in [t], \ t \in [T].$$

Under endogeneous observation of uncertainty, the stage at which the realization of each random variable will be observed is not known, so all NACs must be applied at every stage. Thus the graph structure is the same throughout all stages. By contrast, under exogeneous uncertainty, as we see above, we know a priori how the graph changes over time, so in later stages we don't need a generator for the whole graph.

Indeed, it is not hard to see that if

$$\mathcal{G} \subseteq \bigcup_{t=1}^{T} (\mathcal{T}_t \times \{t\}),$$

enforcing NACs

$$\phi_t^r = \phi_t^s \quad \text{for all } (\{r, s\}, t) \in \mathcal{G}$$

is necessary and sufficient to enforce all NACs,

$$\phi_t^r = \phi_t^s \quad \text{for all } \{r, s\} \in \mathcal{T}_t,\ t \in [T],$$

if and only if $\mathcal{G}$ has the form

$$\mathcal{G} = \bigcup_{t=1}^{T} (\mathcal{G}_t \times \{t\})$$

where $\mathcal{G}_t$ is a spanning subgraph of the connected components of G_t.

As a consequence of this structure, we see that *minimal* sets of NACs, under exogeneous uncertainty, must be precisely those in which each $\mathcal{G}_t$ is a spanning forest for G_t, i.e., a forest such that two vertices are in the same tree of $\mathcal{G}_t$ if and only if they are in the same connected component of G_t. Observe that spanning trees are minimal cardinality spanning subgraphs, and all have the same cardinality, hence are also spanning subgraphs of *minimum cardinality*. In what follows, we shall see that minimal generator sets in the endogeneous case also have this property, since they, like spanning trees, are the bases of a matroid.

In the remainder of this paper, we focus on properties of minimal generators—those with minimal cardinality—in the endogeneous case. To date, reducing the number of NACs applied has focused on scenario sets having an underlying cross product structure, so we examine these first.

4 Minimal generators for cross product scenario sets

In the special case that the $\tilde{\theta}_i$ are discrete random variables, and the scenario set is obtained as a cross product of all realizations, we obtain Goel and Grossmann's Theorem 2 in [14] (also Property 2 in [18]) as a corollary of Theorem 1. When $\{\theta^s : s \in \mathcal{S}\} = \times_{i \in \mathcal{I}} \Theta_i$ (and $\mathcal{D}(r, s) = \{i \in \mathcal{I} : \theta_i^r \neq \theta_i^s\}$), the set $\mathcal{G}^1 := \{\{r, s\} \subseteq \mathcal{S} : |\mathcal{D}(r, s)| = 1\}$ is a generator. To see this we need to check that for every $e = \{r, s\} \in \mathcal{T} \backslash \mathcal{G}^1$ there is a dsc-path for e in $\mathcal{G}$. We prove this by induction on $\eta = |\mathcal{D}(e)|$. For $\eta = 1$ the statement is vacuously true because there is no edge $e \in \mathcal{T} \backslash \mathcal{G}^1$ with $|\mathcal{D}(e)| = 1$. For $\eta \geqslant 2$ with $\mathcal{D}(e) = \{i_1, i_2, \ldots, i_\eta\}$, let s' be the scenario with $\theta_i^{s'} = \theta_i^s$ for $i \neq i_\eta$ and $\theta_{i_\eta}^{s'} = \theta_{i_\eta}^r$. Then $\mathcal{D}(\{r, s'\}) = \{i_1, \ldots, i_{\eta-1}\}$ and $\mathcal{D}(\{s', s\}) = \{i_\eta\}$. By induction, there is a dsc-path P for $e' = \{r, s'\}$ in $\mathcal{G}^1$ with $\mathcal{D}(e'') \subseteq \{i_1, \ldots, i_{\eta-1}\}$ for

all $e'' \in P$, and together with the edge $\{s', s\} \in \mathcal{G}^1$ this forms the required dsc-path for e.

Property 3 of Gupta and Grossmann [18] can also be obtained as a corollary to Theorem 1. Property 3 makes use of an ordering of the set of possible realizations of each random variable to reduce the set of non-anticipativity constraints required in the model. For each $i \in \mathcal{I}$, take $n_i = |\Theta_i|$ and write $\Theta_i = \{h^i_1, \ldots, h^i_{n_i}\}$. With a scenario vector $\theta \in \times_{i\in\mathcal{I}}\Theta_i$ we can associate a vector $\tilde{\theta} \in \times_{i\in\mathcal{I}}\{1, \ldots, n_i\}$ in the obvious way: $\tilde{\theta}_i = j$ for the unique $j \in \{1, \ldots, n_i\}$ with $\theta_i = h^i_j$. Then we can define a distance $d(\theta, \theta')$ between two vectors θ and θ' by $d(\theta, \theta') = \sum_{i\in\mathcal{I}}|\tilde{\theta}_i - \tilde{\theta}'_i|$, and Property 3 states that it is sufficient to require non-anticipativity for scenario pairs of distance 1. We show that such scenario pairs form a generator, and furthermore constitute a minimal generator.

Proposition 2 *If* $\{\theta^s : s \in \mathcal{S}\} = \times_{i\in\mathcal{I}}\Theta_i$ *where* $\Theta_i = \{h^i_1, \ldots, h^i_{n_i}\}$ *for each* $i \in \mathcal{I}$ *and* $\mathcal{C} := \{\{r, s\} \in \mathcal{T} \ : \ d(\theta^s, \theta^r) = 1\}$ *then* $\mathcal{C}$ *is a minimal generator.*

In fact this result follows from a more general one that we give shortly. As a means of introducing the reader to the idea before abstracting it further, we provide a direct proof (in the appendix) and an example to illustrate the concept. Suppose $|\mathcal{I}| = 2$, $\Theta_1 = \{0, 2, 5, 6, 9, 13\}$ and $\Theta_2 = \{3, 4, 5, 7, 10\}$, and consider the scenario pair $\{r, s\} \in \mathcal{T}$ with $\theta^r = (0, 10) = (h^1_1, h^2_5)$ and $\theta^s = (6, 5) = (h^1_4, h^2_3)$. Then the scenario sequence with realization vectors

$$\theta^r = (0, 10), (2, 10), (5, 10), (6, 10), (6, 7), (6, 5) = \theta^s,$$

or equivalently

$$\theta^r = (h^1_1, h^2_5), (h^1_2, h^2_5), (h^1_3, h^2_5), (h^1_4, h^2_5), (h^1_4, h^2_4), (h^1_4, h^2_3) = \theta^s,$$

in which each uncertain parameter in the differentiator set $\mathcal{D}(r, s) = \{1, 2\}$ steps one by one through its realization set between its values under r and s, has consecutive scenarios differing in exactly one element of $\mathcal{D}(r, s)$, with realizations adjacent in that element's realization set order, and hence lie in $\mathcal{C}$, as required for $\mathcal{C}$ to be a generator set.

We now show that these results can be generalized. To do so, it is more convenient to work directly with the realization index sets than the realization values themselves. Thus if $\mu = |\mathcal{I}|$ is the number of sources of uncertainty, we identify the scenario set with

$$\mathcal{S} = \times_{i\in\{1,\ldots,\mu\}}\{1, 2, \ldots, n_i\}.$$

(Recall $n_i = |\Theta_i|$ is the number of possible realizations for $\tilde{\theta}_i$.) The differentiator sets are then given by $\mathcal{D}(r, s) = \{i \in [\mu] \ : \ r_i \neq s_i\}$. For a subset $\mathcal{G} \subseteq \mathcal{T}$, an uncertainty source $i \in [\mu]$, and a vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_\mu) \in \mathcal{S}^*_i$ where $\mathcal{S}^*_i$ is defined by

$$\mathcal{S}^*_i := \left(\times_{j\in\{1,\ldots,i-1\}}[n_j]\right) \times \left(\times_{j\in\{i+1,\ldots,\mu\}}[n_j]\right),$$

we define a graph $G^{\mathcal{G}}_{i,\boldsymbol{\alpha}} := (V_i,\ E^{\mathcal{G}}_{i,\boldsymbol{\alpha}})$ with vertex set $V_i = [n_i]$ and edge set

$$E^{\mathcal{G}}_{i,\boldsymbol{\alpha}} = \{\{p, q\} \ : \ \{(\alpha_1, \ldots, \alpha_{i-1}, p, \alpha_{i+1}, \ldots, \alpha_\mu), \\ (\alpha_1, \ldots, \alpha_{i-1}, q, \alpha_{i+1}, \ldots, \alpha_\mu)\} \in \mathcal{G}\}.$$

Definition 5 We call $\mathcal{G} \subseteq \mathcal{T}$ a *tree conglomerate* if the following two conditions are satisfied.

1. $|\mathcal{D}(r, s)| = 1$ for every $\{r, s\} \in \mathcal{G}$.
2. $G^{\mathcal{G}}_{i,\boldsymbol{\alpha}}$ is a spanning tree on V_i for every $i \in [\mu]$ and every $\boldsymbol{\alpha} \in \mathcal{S}^*_i$.

Theorem 2 *An edge set $\mathcal{G} \subseteq \mathcal{T}$ is a minimal generator if and only if it is a tree conglomerate.*

The proof that tree conglomerates are generators will be by induction on the distance between scenarios r and s. In order to define this distance consider the graph $G^{\mathcal{G}}_{r,s}$, defined to be the subgraph of $(\mathcal{S},\ \mathcal{G})$ induced by the scenario set

$$\mathcal{S}_{r,s} = \{p \in \mathcal{S} \ : \ p_i = r_i = s_i \text{ for all } i \in [\mu] \backslash \mathcal{D}(r, s)\}.$$

Thus any edge $\{p, q\}$ in $G^{\mathcal{G}}_{r,s}$ has $\mathcal{D}(p, q) \subseteq \mathcal{D}(r, s)$, and any r-s-path in $G^{\mathcal{G}}_{r,s}$ must be a dsc-path for $\{r, s\}$. Observe that the second condition of Definition 5 ensures that for any $r, s \in \mathcal{S}$ there exists an r-s-path in $G^{\mathcal{G}}_{r,s}$. Thus when $\mathcal{G}$ is a tree conglomerate, we may define the distance $d^{\mathcal{G}}(r, s)$ to be the distance between r and s in the graph $G^{\mathcal{G}}_{r,s}$, i.e. the number of edges in the r-s-path in $G^{\mathcal{G}}_{r,s}$ with fewest edges. Note that for every $\mathcal{G}$ which satisfies the first condition in Definition 5, we have $\mathcal{G} = \{\{r, s\} \ : \ d^{\mathcal{G}}(r, s) = 1\}$.

Proof First let $\mathcal{G}$ be a tree conglomerate. We prove that for every $e = \{r, s\} \in \mathcal{T} \backslash \mathcal{G}$ there is a dsc-path for e in $\mathcal{G}$. We proceed by induction on $d^{\mathcal{G}}(r, s)$. For $d^{\mathcal{G}}(r, s) = 1$ the statement is vacuously true because there is no edge $e = \{r, s\} \in \mathcal{T} \backslash \mathcal{G}$ with $d^{\mathcal{G}}(r, s) = 1$. For $d^{\mathcal{G}}(r, s) \geqslant 2$, let s' be the last scenario before s on a shortest r-s-path in $G^{\mathcal{G}}_{r,s}$. Then $d^{\mathcal{G}}(r,\ s') = d^{\mathcal{G}}(r,\ s) - 1$ and $d^{\mathcal{G}}(s', s) = 1$, i.e., $\{s', s\} \in \mathcal{G}$. By induction, there is a dsc-path P for $e' = \{r, s'\}$ in $\mathcal{G}$ with $\mathcal{D}(e'') \subseteq \mathcal{D}(e') \subseteq \mathcal{D}(e)$, and together with the edge $\{s', s\}$ we obtain the required dsc-path for e.

For the minimality of $\mathcal{G}$, consider $e = \{r, s\} \in \mathcal{G}$:

$$r = (\alpha_1, \ldots, \alpha_{i-1}, p, \alpha_{i+1}, \ldots, \alpha_\mu), \quad s = (\alpha_1, \ldots, \alpha_{i-1}, q, \alpha_{i+1}, \ldots, \alpha_\mu).$$

Any dsc-path for e can contain only edges e' with $\mathcal{D}(e') = \{i\}$. So any dsc-path for e (other than e itself) corresponds to a p-q-path in $(V_i, E^{\mathcal{G}}_{i,\boldsymbol{\alpha}} \backslash \{\{p, q\}\})$, which cannot exist because $G^{\mathcal{G}}_{i,\boldsymbol{\alpha}}$ is a spanning tree for V_i. This shows that $\mathcal{G} \backslash \{e\}$ is not a generator.

For the converse, let $\mathcal{G}$ be a minimal generator. We need to establish the following three claims.

Claim 1. For every $i \in [\mu]$ and every $\boldsymbol{\alpha} \in \mathcal{S}^*_i$, the graph $G^{\mathcal{G}}_{i,\boldsymbol{\alpha}}$ is connected. This true because if any vertices $p, q \in V_i$ cannot be connected by a path in $G^{\mathcal{G}}_{i,\boldsymbol{\alpha}}$, the edge $e = \{r, s\}$ for scenarios

$$r = (\alpha_1, \ldots, \alpha_{i-1}, p, \alpha_{i+1}, \ldots, \alpha_\mu), \quad s = (\alpha_1, \ldots, \alpha_{i-1}, q, \alpha_{i+1}, \ldots, \alpha_\mu)$$

is not in $\mathcal{G}$, and there is no dsc-path for e in $\mathcal{G}$, contradicting $\mathcal{G}$ a generator.

Claim 2. For every $i \in [\mu]$ and every $\boldsymbol{\alpha} \in \mathcal{S}_i^*$, the graph $G_{i,\boldsymbol{\alpha}}^{\mathcal{G}}$ is cycle-free. This is true because a cycle in $G_{i,\boldsymbol{\alpha}}^{\mathcal{G}}$ corresponds to a cycle C in $\mathcal{G}$ with $\mathcal{D}(e') = \{i\}$ for all $e' \in C$. Thus for any $e \in C$, $\mathcal{G}\backslash\{e\}$ is still a generator, contradicting minimality of $\mathcal{G}$.

Claim 3. If $e = \{r, s\} \in \mathcal{G}$ with $|\mathcal{D}(e)| > 1$ then $\mathcal{G}\backslash\{e\}$ is still a generator. Let $\mathcal{D}(e) = \{i_1, \ldots, i_k\}$. By Claim 1, $\mathcal{G}$ contains a path P_1 from r to r^1 with

$$r_i^1 = \begin{cases} r_i & \text{for } i \neq i_1, \\ s_i & \text{for } i = i_1 \end{cases}$$

and $\mathcal{D}(e') = \{i_1\}$ for all $e \in P_1$. Then Claim 1 yields the existence of a path P_2 in $\mathcal{G}$ from r_1 to r_2 with

$$r_i^2 = \begin{cases} r_i^1 & \text{for } i \neq i_2, \\ s_i & \text{for } i = i_1 \end{cases}$$

and $\mathcal{D}(e') = \{i_2\}$ for all $e \in P_2$. Continuing in this way we obtain a sequence of paths $P_1, \ldots, P_k$ with $\mathcal{D}(e') = \{i_h\}$ for $h = 1, 2, \ldots, k$. Concatenating the paths $P_1, \ldots, P_k$ we obtain a dsc-path for e in $\mathcal{G}\backslash\{e\}$, thus establishing that e can be removed.

Claims 1 and 2 establish the second condition of Definition 5; Claim 3 establishes its first condition. Hence $\mathcal{G}$ must be a tree conglomerate. □

Remark 1 Set $\mathcal{C}$ from Proposition 2 (also Property 3 of [18]) is the special case of a tree conglomerate where $G_{i,\boldsymbol{\alpha}}^{\mathcal{G}}$ is the path $(1, 2, \ldots, n_i)$ for every i and $\boldsymbol{\alpha}$. Thus Proposition 2 follows as a corollary of Theorem 2. Note that since $\mathcal{G}^1 \supseteq \mathcal{C}$, this also shows that $\mathcal{G}^1$ is a generator set, hence by Theorem 1 nonanticipativity constraints in $\mathcal{G}^1$ suffice for validity ([18] Property 2). Of course $\mathcal{G}^1$ is not a minimal NAC-sufficient set, whereas $\mathcal{C}$ is. A slightly more general special case of a tree conglomerate is to fix spanning trees T_i on the vertex sets V_i and take the cartesian product of the trees T_i. This will yield tree conglomerates with the same cardinality as $\mathcal{C}$. (Our results in the next section will show this is to be expected in general, as all minimal generators are bases of the same matroid and hence must have the same cardinality.)

The benefits of generator sets in terms of the reduction in number of nonanticipativity constraints required is readily observed in the case of cross product constructed scenario sets. From

$$|\mathcal{T}| = \frac{1}{2} \prod_{i \in \mathcal{I}} n_i \left(\prod_{j \in \mathcal{I}} n_j - 1 \right)$$

we get reduction to

$$|\mathcal{G}^1| = \sum_{j\in\mathcal{I}} \prod_{i\in\mathcal{I}\setminus\{j\}} n_i \left(\frac{1}{2} n_j (n_j - 1)\right)$$

(formed by taking each $j \in \mathcal{I}$ and each realization vector in $\times_{i\in\mathcal{I}\setminus\{j\}}\Theta_i$, and then forming all distinct pairs of elements in Θ_j), and then further reduction (eliminating the $\frac{1}{2} n_j$ term) to

$$|\mathcal{C}| = \sum_{j\in\mathcal{I}} \prod_{i\in\mathcal{I}\setminus\{j\}} n_i (n_j - 1)$$

(formed by taking each $j \in \mathcal{I}$ and each realization vector in $\times_{i\in\mathcal{I}\setminus\{j\}}\Theta_i$, and then forming all pairs of adjacent elements in Θ_j, giving $n_j - 1$ pairs). As noted by Gupta and Grossmann [18], in the case that $|\Theta_i| = K$ for all $i \in \mathcal{I}$, this gives

$$|\mathcal{T}| = \frac{1}{2} K^\mu (K^\mu - 1), \quad |\mathcal{G}^1| = \frac{1}{2} \mu K^\mu (K - 1) \text{ and } |\mathcal{C}| = \mu K^{\mu-1} (K - 1).$$

As illustrated in [18], for $K = 3$ and $\mu = 2$, this gives a reduction from $|\mathcal{T}| = 36$ to $|\mathcal{G}^1| = 18$ to $|\mathcal{C}| = 12$ nonanticipativity constraints required. The effects are of course even more dramatic as the number of realizations and sources of endogeneous uncertainty rise, e.g. for $K = 5$ and $\mu = 2$, we have $|\mathcal{T}| = 29{,}403$, $|\mathcal{G}^1| = 1215$ to $|\mathcal{C}| = 810$.

The above discussion concerns generator sets for scenario sets formed from all cross products of the possible realizations of the endogeneous uncertain parameters. How can we find generators in general? Naturally we would seek a minimal generator; ideally it would be one of minimum cardinality. In fact, as we will shortly prove, a greedy algorithm will find just that.

5 Greedy is optimal for minimizing cardinality of generator sets

In what follows, we will need to manipulate dsc-paths used in the definition of generator sets. We first make a helpful observation.

Remark 2 Let $\mathcal{G}$ be a generator and suppose $e = \{r, s\} \in \mathcal{G}$ has a dsc-path, P, in $\mathcal{G}\setminus\{e\}$. In other words, suppose there exists P an r-s-path with $\mathcal{D}(e') \subseteq \mathcal{D}(e)$ for all $e' \in P$ and $P \subseteq \mathcal{G}\setminus\{e\}$. Now for any $e' = \{r', s'\} \in \mathcal{T}\setminus\{e\}$ one of two cases must hold: either (i) $\mathcal{D}(e') \nsubseteq \mathcal{D}(e)$ so any dsc-path for e' in $\mathcal{G}$ does not include e, or (ii) for any dsc-path, P', for e' in $\mathcal{G}$ using e, the path

$$(P'\setminus\{e\}) \cup P$$

is a dsc-path for e' in $\mathcal{G}\setminus\{e\}$ (since differentiator set containment is transitive). Thus $\mathcal{G}\setminus\{e\}$ is also a generator.

To show that any minimal generator is in fact one of minimum cardinality, and hence that a greedy algorithm can be used to find a generator of minimum cardinality, we prove that the collection of sets of scenario pairs that are the complement of a generator set is a matroid. We first recall the definition of a matroid, in terms of the notation in which we will apply it.

Definition 6 A *matroid* consists of a finite set of elements, $\mathcal{T}$, together with a collection of subsets of $\mathcal{T}$, denoted by Φ, that satisfies the following properties:

1. Φ contains the empty set,
2. for $F \in \Phi$ and $F' \subseteq F$, it must be that $F' \in \Phi$, and
3. if $F_1, F_2 \in \Phi$ with $|F_1 \backslash F_2| = 1$ and $|F_2 \backslash F_1| = 2$, then there must be some $e \in F_2 \backslash F_1$ such that $F_1 \cup \{e\} \in \Phi$.

Note that the third property is more usually stated as

> if $F_1, F_2 \in \Phi$ with $|F_1| < |F_2|$, then there must be some $e \in F_2 \backslash F_1$ such that $F_1 \cup \{e\} \in \Phi$.

(See, for example, the text by Schrijver [27].) The definition we give above is equivalent, and somewhat more convenient for our use.

Theorem 3 *The collection $\Phi = \{\mathcal{T} \backslash \mathcal{G} : \mathcal{G}$ is a generator$\}$ is a matroid on the ground set $\mathcal{T}$.*

Proof The collection Φ is not empty because $\emptyset \in \Phi$. If $F' \subseteq F$ and $F \in \Phi$ then $\mathcal{T} \backslash F' \supseteq \mathcal{T} \backslash F$ is a generator, and consequently $F' \in \Phi$. Thus the first two properties required of a matroid are satisfied. To check the third, we suppose that $F_1 \backslash F_2 = \{e^*\}$ and $F_2 \backslash F_1 = \{e_1, e_2\}$ with $e^* = \{x, y\}$ and $e_i = \{x_i, y_i\}$ for $i \in \{1, 2\}$. There are x_i-y_i-paths P_i in $\mathcal{T} \backslash F_2$ such that $\mathcal{D}(e') \subseteq \mathcal{D}(e_i)$ for every $e' \in P_i$. If $e^* \notin P_i$ then P_i witnesses that e_i can be removed from the generator set $\mathcal{T} \backslash F_1$ without losing the generator property, and consequently $F_1 \cup \{e_i\} \in \Phi$. Assume $e^* \in P_i$. There is an x-y-path P in $\mathcal{T} \backslash F_1$ with $\mathcal{D}(e') \subseteq \mathcal{D}(e^*)$ for every $e' \in P$. If $e_i \notin P$ then we can replace e^* by P to obtain the x_i-y_i-path

$$P_i' = \left(P_i \backslash \{e^*\}\right) \cup P$$

in $\mathcal{T} \backslash (F_1 \cup \{e_i\})$ which ensures that $F_1 \cup \{e_i\} \in \Phi$. The remaining case $e_i \in P$ can occur only if $\mathcal{D}(e^*) = \mathcal{D}(e_i)$. This implies that we may assume $e^* \in P_1 \cap P_2$ and $\mathcal{D}(e^*) = \mathcal{D}(e_1) = \mathcal{D}(e_2)$. Without loss of generality, P_1 has the vertex sequence

$$x_1 = v_0, v_1, \ldots, v_k, v_{k+1}, v_{k+2}, \ldots, v_m = y_1,$$

where $v_k = x$ and $v_{k+1} = y$, and P_2 has the form

$$x_2 = w_0, w_1, \ldots, w_{k'}, w_{k'+1}, w_{k+2}, \ldots, w_{m'} = y_2,$$

where $w_{k'} = x$ and $w_{k'+1} = y$. Now the path

$$x_1 = v_0, v_1, \ldots, v_k = x = w_{k'}, w_{k'-1}, w_{k'-2}, \ldots, w_0 = x_2, y_2 = w_{m'}, w_{m'-1}, \ldots, w_{k'+1} = y = v_{k+1}, v_{k+2}, \ldots, v_m = y_1$$

ensures that $F \cup \{e_1\} \in \Phi$. □

We have thus proved the following for generally defined endogenous uncertainty scenario sets.

Corollary 1 *Any minimal generator of $\mathcal{T}$ is one of minimum cardinality.*

Consequently a minimum cardinality generator $\mathcal{G}^*$ of $\mathcal{T}$ can be found by a greedy algorithm, for example, by initializing $\mathcal{G}^* := \mathcal{T}$ and then iteratively removing elements from $\mathcal{G}^*$ while retaining the property that $\mathcal{G}^*$ is a generator of $\mathcal{T}$. The straightforward implementation gives a runtime of $\mathcal{O}(|\mathcal{S}|^4)$: for each $e = \{r, s\} \in \mathcal{T}$ check if there is a dsc-path for e in $\mathcal{G}^* \backslash \{e\}$, and remove e from $\mathcal{G}^*$ if that is the case.

We illustrate these concepts with a small example, having three sources of uncertainty, given by the random variables $\tilde{\theta}_1$, $\tilde{\theta}_2$ and $\tilde{\theta}_3$, each with three possible realizations. For ease of exposition, we denote these by $\Theta_i = \{L, M, H\}$ (e.g. representing "Low", "Medium" and "High" values) for $i = 1, 2, 3$ and simply concatenate the symbols to form a scenario, i.e. write MLL rather than (M, L, L). Only 7 of the 27 possible cross products are valid scenarios: the scenarios are

$$\mathcal{S} = \{LLL, LLM, LMH, LHM, MLL, MMM, HHL\}.$$

Although we may initialize $\mathcal{G}$ to be all 21 unordered pairs of scenarios in $\mathcal{T}$, in fact we observe that all pairs $\{r, s\}$ with $\mathcal{D}(r, s) = \{1, 2, 3\} = \mathcal{I}$ can immediately be removed, since the result is a connected graph on $\mathcal{S}$: every pair of scenarios $\{r, s\}$ must be connected by a path in the graph, and since *all* differentiator sets are subsets of $\mathcal{I}$, it must be that every remaining edge must have differentiator set a subset of $\mathcal{D}(r, s)$. Thus without too much effort, we may start with $\mathcal{G}$ taken to be the pairs indicated in Fig. 3.

We now see if any pairs in $\mathcal{G}$ can be removed without losing the generator property. For example, consider the pair $\{LLL, LHM\}$ with $\mathcal{D}(LLL, LHM) = \{2, 3\}$. Observe that if there is a dsc-path for $\{LLL, LHM\}$ in $\mathcal{G}$ distinct from the edge $\{LLL, LHM\}$ (call this path P) then we may safely remove it: the path P maintains the connectivity between LLL and LHM, and any other pair that uses $\{LLL, LHM\}$ in its dsc-path can have that edge replaced in that path by P, as per Remark 2. Now $\{\{LLL, LLM\}, \{LLM, LHM\}\}$ is a dsc-path for $\{LLL, LHM\}$ in $\mathcal{G}$, since $\mathcal{D}(LLL, LLM) \cup \mathcal{D}(LLM, LHM) = \{3\} \cup \{2\} = \{2, 3\} \subseteq \mathcal{D}(LLL, LHM)$. Thus we can replace $\mathcal{G} := \mathcal{G} \backslash \{\{LLL, LHM\}\}$ and be sure $\mathcal{G}$ is still a generator. By next considering each of the pairs shown as dashed edges in Fig. 3 in turn, we find that all can be removed in a similar fashion, resulting in $\mathcal{G}$ consisting of the solid edges in the graph. Attempting to remove any of the pairs corresponding to solid edges does not succeed: in all cases the corresponding edge is the only dsc-path for its pair in the

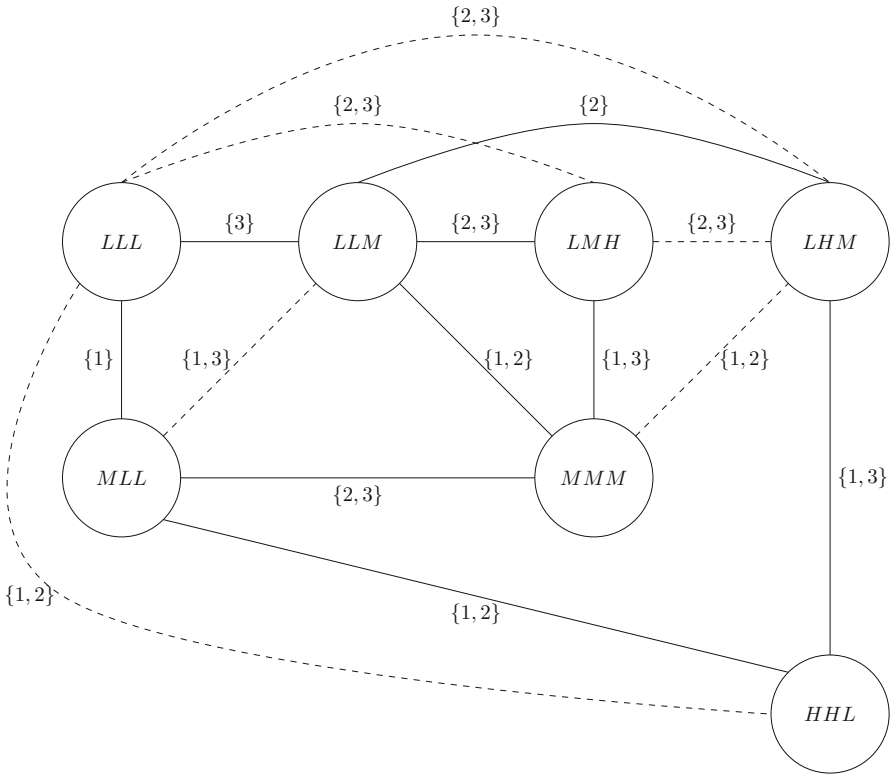

Fig. 3 The graph $(\mathcal{S}, \mathcal{G})$ for the example, showing the initial generator $\mathcal{G}$ (all edges) after all pairs with differentiator set {1, 2, 3} have been removed, and the final generator (*solid edges* only). The edge label for edge $\{r, s\}$ is the differentiator set $\mathcal{D}(r, s)$ for the pair of scenarios $\{r, s\}$

generator. Thus the generator $\mathcal{G}$ given by the solid edges is minimal, and hence by Corollary 1 is a sufficient set of minimum cardinality.

Note that, depending on the order in which pairs are considered for removal, different generator sets may result. For example, if $\{LLM, LMH\}$ is considered first, rather than $\{LLL, LMH\}$ then the latter will replace the former in the minimal generator. However regardless of the order used, Corollary 1 ensures the end result is a minimum cardinality generator.

To conclude, we observe that minimum cardinality generators can also be constructed "upward", starting from the empty set, rather than "downward", starting from $\mathcal{T}$.

Remark 3 The dual of Φ is the matroid

$$\Phi^* = \{F \subseteq \mathcal{T} \ : \ F \text{ is contained in some minimal generator}\}.$$

Minimal generators are precisely the bases of the matroid Φ^*, and we can determine one in the following way. We order the elements $e \in \mathcal{T}$ in non-decreasing order of the cardinalities of the sets $\mathcal{D}(e)$:

$$|\mathcal{D}(e_1)| \leqslant |\mathcal{D}(e_2)| \leqslant \cdots \leqslant |\mathcal{D}(e_m)|.$$

We initialize $\mathcal{G} = \emptyset$. For $i = 1, 2, \ldots, m$, if $\mathcal{G}$ does not contain a dsc-path for e_i then we add e_i to $\mathcal{G}$. The final $\mathcal{G}$ is a basis of the matroid Φ^*, i.e., a minimal generator.

Illustrating this approach on the small example, we see that the ordering

$$e_1 = \{LLL, MLL\}, e_2 = \{LLM, LHM\}, e_3 = \{LLL, LLM\},$$
$$e_4 = \{MLL.HHL\}, e_5 = \{LLM, MMM\}, e_6 = \{LMH, MMM\},$$
$$e_7 = \{LHM, HHL\}, e_8 = \{LLM, LMH\}, e_9 = \{MLL, MMM\}, \ldots \text{ etc.}$$

yields the minimal generator given by the solid lines in Fig. 3.

6 Conclusions

This paper contributes to the general field of multistage stochastic programming with endogenous uncertainty by characterizing necessary and sufficient sets of non-anticipativity constraints, without any restriction on the scenario space. Existing results for scenario sets constructed from cross products are obtained as corollaries of the results presented here, and generalized. We prove that sufficient sets of non-anticipativity constraints have matroid structure, and hence prove that such sets having minimum cardinality can be identified efficiently, in the general case.

Acknowledgments The authors are very grateful to BHP Billiton Ltd. and in particular to Merab Menabde, Peter Stone, and Mark Zuckerberg for their support of the mining-related research that inspired this work. We also thank Hamish Waterer and Laana Giles for useful discussions in the early stages of the research, and thank Hamish for his helpful suggestions and proof-reading of early versions of this paper. We are most grateful to Ignacio Grossmann for his advice and encouragement in completing the paper. This research would not have been possible without the support of the Australian Research Council, grant LP0561744. Finally, the efforts of two anonymous reviewers in improving the paper were greatly appreciated.

Appendix

This appendix provides the proofs of Propositions 1 and 2.

Proof (Proof of Proposition 1) Let $\{r, s\} \in \mathcal{T}$, let $\sigma \in \{r, s\}$ be chosen so that (8) holds, and define $\bar{\sigma}$ so that $\{\sigma, \bar{\sigma}\} = \{r, s\}$. Thus (8) ensures that, for any given $i \in \mathcal{I}$,

$$\sum_{j \in \mathcal{D}(r,s)} \beta^{\sigma}_{j,t} = 0 \;\Rightarrow\; \beta^{s}_{i,t+1} = \beta^{r}_{i,t+1}, \;\forall t = 1, \ldots, T-1 \tag{9}$$

is satisfied. To show (6) holds, we must show

$$\sum_{j \in \mathcal{D}(r,s)} \beta^{\bar{\sigma}}_{j,t} = 0 \;\Rightarrow\; \beta^{s}_{i,t+1} = \beta^{r}_{i,t+1}, \;\forall t = 1, \ldots, T-1 \tag{10}$$

is also satisfied. We proceed by induction on t. For the case $t = 1$, (7) ensures $\beta^{\sigma}_{i,1} = 0$ so by (9) we have $\beta^{s}_{i,2} = \beta^{r}_{i,2}$, and (10) must hold for $t = 1$. Make the inductive assumption that (10) holds for some $t \in \{1, \ldots, T-2\}$. Now suppose that $\sum_{j \in \mathcal{D}(r,s)} \beta^{\tilde{\sigma}}_{j,t+1} = 0$. Then by (2), $\sum_{j \in \mathcal{D}(r,s)} \beta^{\tilde{\sigma}}_{j,t} = 0$, so by the inductive assumption, it must be that $\beta^{s}_{i,t+1} = \beta^{r}_{i,t+1}$. Hence $\sum_{j \in \mathcal{D}(r,s)} \beta^{\sigma}_{j,t+1} = 0$ and so by (9) it must be that $\beta^{s}_{i,t+2} = \beta^{r}_{i,t+2}$ as required; (10) holds for $t+1$. □

Proof (Proof of Proposition 2) We prove that $\mathcal{C}$ is a generator by showing that for every $e = \{r, s\} \in \mathcal{T}\backslash\mathcal{C}$ there is a dsc-path for e in $\mathcal{C}$. We proceed by induction on $d(\theta^r, \theta^s)$. For $d(\theta^r, \theta^s) = 1$ the statement is vacuously true because there is no edge $e = \{r, s\} \in \mathcal{T}\backslash\mathcal{C}$ with $d(\theta^r, \theta^s) = 1$. For $d(\theta^r, \theta^s) \geqslant 2$, pick $i \in \mathcal{I}$ with $\theta^r_i \neq \theta^s_i$, let j and j' be the indices with $\theta^r_i = h^i_j$ and $\theta^s_i = h^i_{j'}$, and define s' by $\theta^{s'}_k = \theta^{s'}_k$ for $k \neq i$ and

$$\theta^{s'}_i = \begin{cases} h^i_{j'+1} & \text{if } j > j', \\ h^i_{j'-1} & \text{if } j < j'. \end{cases}$$

Then $d(\theta^r, \theta^{s'}) = d(\theta^r, \theta^s) - 1$ and $d(\theta^{s'}, \theta^s) = 1$, i.e., $\{s', s\} \in \mathcal{C}$. By induction, there is a dsc-path P for $e' = \{r, s'\}$ in $\mathcal{C}$ with $\mathcal{D}(e'') \subseteq \mathcal{D}(e') \subseteq \mathcal{D}(e)$, and together with the edge $\{s', s\}$ we obtain the required dsc-path for e.

Minimality of $\mathcal{C}$ is not difficult to establish. Consider $e = \{r, s\} \in \mathcal{C}$: without loss of generality suppose $\theta^r = (h^1_k, \alpha_2, \ldots, \alpha_\mu)$ (where $\mu = |\mathcal{I}|$) and $\theta^s = (h^1_{k+1}, \alpha_2, \ldots, \alpha_\mu)$ for some $(\alpha_2, \ldots, \alpha_\mu) \in \times\Theta_{\mathcal{I}\backslash\{1\}}$ and some $k \in \{1, \ldots, n_1 - 1\}$. Then $\mathcal{D}(e) = \{1\}$ and any dsc-path for e in $\mathcal{C}$ must have the form $r = v_0, v_1, \ldots, v_m = s$ with $e_j = \{v_{j-1}, v_j\} \in \mathcal{C}$ and $\mathcal{D}(e_j) = \{1\}$ for $j = 1, \ldots, m$. Thus it must be that $\theta^{v_j} = (h^1_{i_j}, \alpha_2, \ldots, \alpha_\mu)$ for $h^1_{i_j} \in \Theta_1$ and $|i_j - i_{j+1}| = 1$, for all j. So $i_0 = k$, $i_m = k + 1$, $i_j \in \{1, \ldots, n_1\}$ for all $j = 0, \ldots m$ and $|i_j - i_{j-1}| = 1$, for all $j = 1, \ldots, m$. Since the path can be assumed to be simple, $\{i_j\}_{j=0}^m$ are distinct. The unique solution to these requirements is to take $m = 1$. Thus $\{r, s\}$ is necessary; if it is removed from $\mathcal{C}$, the set will no longer be a generator. □

References

1. Artstein, Z., Wets, R.J.-B.: Sensors and information in optimization under stochastic uncertainty. Math. Oper. Res. **28**, 523–547 (1993)
2. Asamov, T., Ruszczyński, A.: Time-consistent approximations of risk-averse multistage stochastic optimization problems. Math. Progr. 1–35 (2014). doi:10.1007/s10107-014-0813-x
3. Bertsimas, D., Georghiou, A.: Design of near optimal decision rules in multistage adaptive mixed-integer optimization. Oper. Res. 1–18 (2015). doi:10.1287/opre.2015.1365
4. Birge, J.R., Louveaux, F.: Introduction to Stochastic Programming. Springer, Berlin (1997)
5. Boland, N., Dumitrescu, I., Froyland, G.: A multistage stochastic programming approach to open pit mine production scheduling with uncertain geology. Optim. Online (2008). http://www.optimization-online.org/DB_FILE/2008/10/2123.pdf
6. Bruni, M.E., Beraldi, P., Conforti, D.: A stochastic programming approach for operating theatre scheduling under uncertainty. IMA J. Manag. Math. **26**(1), 99–119 (2014)
7. Colvin, M., Maravelias, C.T.: A stochastic programming approach for clinical trial planning in new drug development. Comput. Chem. Eng. **32**, 2626–2642 (2008)

8. Colvin, M., Maravelias, C.T.: Scheduling of testing tasks and resource planning in new product development using stochastic programming. Comput. Chem. Eng. **33**(5), 964–976 (2009)
9. Colvin, M., Maravelias, C.T.: Modeling methods and a branch and cut algorithm for pharmaceutical clinical trial planning using stochastic programming. Eur. J. Oper. Res. **203**, 205–215 (2010)
10. Fragnière, E., Gondzio, J., Yang, X.: Operations risk management by optimally planning the qualified workforce capacity. Eur. J. Oper. Res. **202**, 518–527 (2010)
11. Georghiou, A., Wiesemann, W., Kuhn, D.: Generalized decision rule approximations for stochastic programming via liftings. Math. Progr. 1–38 (2014). doi:10.1007/s10107-014-0789-6
12. Giles, L.: A Multi-stage Stochastic Model for Hydrogeological Optimisation, Masters Thesis, Department of Engineering Science, The University of Auckland (2009)
13. Goel, V., Grossmann, I.E.: A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves. Comput. Chem. Eng. **28**(8), 1409–1429 (2004)
14. Goel, V., Grossmann, I.E.: A class of stochastic programs with decision dependent uncertainty. Math. Progr. Ser. B **108**, 355–394 (2006)
15. Goel, V., Grossmann, I.E., El-Bakry, A.S., Mulkay, E.L.: A novel branch and bound algorithm for optimal development of gas fields under uncertainty in reserves. Comput. Chem. Eng. **30**, 1076–1092 (2006)
16. Guigues, V., Sagastizábal, C.: Risk-averse feasible policies for large-scale multistage stochastic linear programs. Math. Program. **138**, 167–198 (2013)
17. Gupta, V.: Modeling and Computational Strategies for Optimal Oilfield Development Planning Under Fiscal Rules and Endogenous Uncertainties, PhD Thesis, Carnegie Mellon University (2013)
18. Gupta, V., Grossmann, I.E.: Solution strategies for multistage stochastic programming with endogenous uncertainties. Comput. Chem. Eng. **35**, 2235–2247 (2011)
19. Gupta, V., Grossmann, I.E.: A new decomposition algorithm for multistage stochastic programs with endogenous uncertainties. Comput. Chem. Eng. **62**, 62–79 (2014)
20. Higle, J.L., Rayco, B., Sen, S.: Stochastic scenario decomposition for multistage stochastic programs. IMA J. Manag. Math. **21**(1), 39–66 (2010)
21. Jonsbråten, T.W., Wets, R.J.-B., Woodruff, D.L.: A class of stochastic programs with decision dependent random elements. Ann. Oper. Res. **82**, 83–106 (1998)
22. Kozmík, V., Morton, D.P.: Evaluating policies in risk-averse stochastic dual dynamic programming. Math. Progr. 1–26 (2014). doi:10.1007/s10107-014-0787-8
23. Philpott, A.B., de Matos, V.L.: Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. Eur. J. Oper. Res. **218**, 470–483 (2012)
24. Ramazan, S., Dimitrakopoulos, R.: Production scheduling with uncertain supply: a new solution to the open pit mining problem. Optim. Eng **14**, 361–380 (2013)
25. Ruszczyński, A.: Decomposition methods. In: Ruszczyński, A., Shapiro, A. (eds.) Stochastic Programming, Handbook in OR & MS, vol. 10. North-Holland Publishing Company, Amsterdam (2003)
26. Sahinidis, N.V.: Optimization under uncertainty: state-of-the-art and opportunities. Comput. Chem. Eng. **28**(6–7), 971–983 (2004)
27. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency, Volume B. Springer, Berlin (2003)
28. Schultz, R.: Stochastic programming with integer variables. Math. Program. **97**(12), 285–309 (2003)
29. Sen, S., Zhou, Z.: Multistage stochastic decomposition: a bridge between stochastic programming and approximate dynamic programming. SIAM J. Optim. **24**(1), 127–153 (2014)
30. Shapiro, A.: On complexity of multistage stochastic programs. Oper. Res. Lett. **34**, 1–8 (2006)
31. Shapiro, A.: Analysis of stochastic dual dynamic programming method. Eur. J. Oper. Res. **209**(1), 63–72 (2011)
32. Shapiro, A., Dentcheva, D., Ruszczyski, A.P.: Lectures on Stochastic Programming: Modeling and Theory, Vol. 9. SIAM (2009)
33. Shapiro, A., Tekaya, W., da Costa, J.P., Soares, M.P.: Risk neutral and risk averse stochastic dual dynamic programming method. Eur. J. Oper. Res. **224**(2), 375–391 (2013)
34. Solak, S., Clarke, J.-P.B., Johnson, E.L., Barnes, E.R.: Optimization of R&D project portfolios under endogenous uncertainty. Eur. J. Oper. Res. **207**, 420–433 (2010)
35. Tarhan, B., Grossmann, I.E.: A multistage stochastic programming approach with strategies for uncertainty reduction in the synthesis of process networks with uncertain yields. Comput. Chem. Eng. **32**, 766–788 (2008)

36. Tarhan, B., Grossmann, I.E., Goel, V.: Stochastic programming approach for the planning of offshore oil or gas field infrastructure under decision-dependent uncertainty. Ind. Eng. Chem. Res. **48**(6), 3078–3097 (2009)
37. Tarhan, B., Grossmann, I.E., Goel, V.: Computational strategies for non-convex multistage MINLP models with decision-dependent uncertainty and gradual uncertainty resolution. Ann. Oper. Res. **203**(1), 141–166 (2013)
38. Vayanos, P., Kuhn, D., Rustem, B.: Decision rules for information discovery in multi-stage stochastic programming. In: Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC–ECC), Orlando, FL, December 12-15, 2011, pp. 7368–7373