7th International Symposium on Imprecise Probability: Theories and Applications, Innsbruck, Austria, 2011

Dynamic Programming and Subtree Perfectness for Deterministic Discrete-Time Systems with Uncertain Rewards

Nathan Huntley
Durham University, UK
nathan.huntley@durham.ac.uk

Matthias C. M. Troffaes
Durham University, UK
matthias.troffaes@gmail.com

Abstract

We generalise de Cooman and Troffaes's sufficient condition for dynamic programming to work for deterministic discrete-time systems. To do so, we use the general framework developed by Huntley and Troffaes, for decision trees with arbitrary rewards and arbitrary choice functions. Whence, we allow deterministic discrete-time systems with arbitrary rewards and an arbitrary composition operator on rewards. We show that the principle of optimality reduces to two much simpler conditions on the choice function. We establish necessary and sufficient conditions on choice functions for deterministic discrete-time systems to be solvable by backward induction, that is, for dynamic programming to work. Finally, we also discuss subtree perfectness—which is a stronger form of dynamic consistency—for these systems, and show that, in general, decision criteria from imprecise probability theory violate it, even though dynamic programming may work.

Keywords. Optimal control, dynamic programming, deterministic discrete-time systems, backward induction, subtree perfectness, choice function

1 Introduction

In this paper we formalize and extend the results of de Cooman and Troffaes [4] for deterministic discrete-time systems with uncertain gains. Such systems are typical in *control theory* (see for instance [2, 9]), which more generally covers the behaviour and control of dynamic systems. The particular class of systems we investigate is best illustrated by example: Fig. 1 depicts a system that starts at N_1, and can reach N_4 by multiple paths. The subject, who controls the system, can choose the path the system will take. Travelling down a particular arc gives the subject an associated reward. For instance, choosing the arc from N_1 to N_2 will give the subject X. The subject's task is to find an optimal path for the system to take.

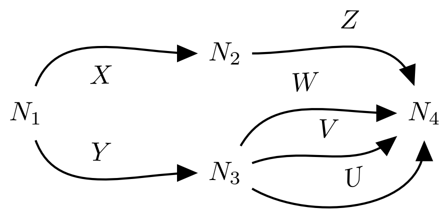

Figure 1: A simple deterministic system.

This is an example of a *deterministic discrete-time system.* If all rewards $U, \ldots, Z$ are certain, so the subject knows exactly what she will receive when choosing a particular route, then this is a system with *certain gains.* Such systems are easily solved: find a path with the highest total reward. We instead consider systems with uncertain gains, so $U, \ldots, Z$ give rewards determined by the as yet unknown state of nature. Such uncertain gains are called *gambles.* The overall reward for a particular path is then determined by the sum of the gambles for all arcs in the path.

This paper deals with *normal form* decision making. In general, normal form decisions involve the subject specifying her decisions in all eventualities, and then acting upon this specification. For deterministic discrete-time systems with certain rewards, a normal form decision is simply a path through the system. In contrast, the *extensive form* involves making decisions only when the relevant decision point is reached, and is expressed differently. We do not investigate the extensive form in this paper, but caution that the two forms do not always lead to the same answer.

With uncertain rewards, there are two possible ways the system can evolve. If the subject receives the reward from a gamble as soon as that arc is chosen, then she can use this information to choose her next arc. For example, an informal strategy for Fig. 1 could be "choose Y, and then choose W if Y has given a large reward, but choose V otherwise". Alternatively, the

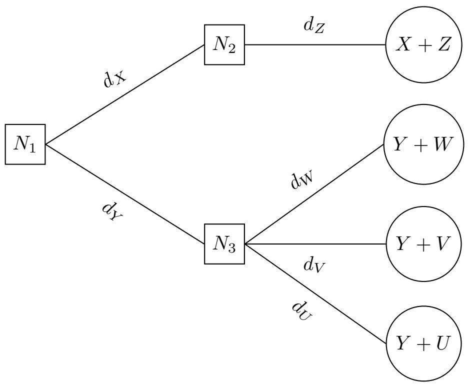

Figure 2: The decision tree for Fig. 1.

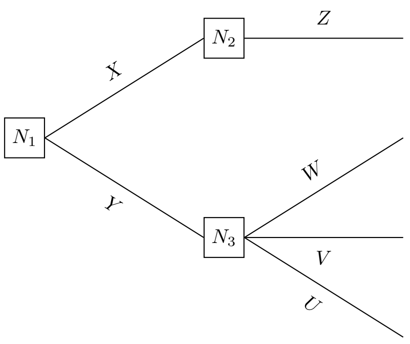

Figure 3: The deterministic system tree for Fig. 1.

subject may only learn about her actual rewards at the end of the process, and so could have no strategy more complicated than, say "choose Y, then W", because she does not learn of the outcome of Y until later. The latter set-up, where the true state of nature is only revealed at the end of the process, is followed by de Cooman and Troffaes, and so we follow it too.

Normal form decisions are thus very simple (indeed, exactly the same as for certain rewards), and no concept of conditioning is required. Also, since in this case everything is completely deterministic until the final decision has been made, it seems natural to use the normal form. Note that the concept of normal form decision making can be criticized [14], however we do not aim to address these issues in this paper.

We aim to apply known results by Huntley and Troffaes [6] on backward induction and subtree perfectness for decision trees to these deterministic discrete-time systems, thereby generalizing the work of de Cooman and Troffaes [4]. Whence, as a first step, we represent these systems as decision trees [8, 7, 3]. An example is given in Fig. 2. In such a tree, square nodes, called decision nodes, represent points at which the subject must choose an arc. The circular nodes, called chance nodes represent points at which the consequence is determined by the state of nature. For completeness, Fig. 3 ought to have arcs leading from the chance nodes to terminal *reward* nodes, representing the rewards given by the gambles for particular states of nature. Since we have not explicitly defined the gambles, this final layer of nodes has been omitted.

This representation can be simplified to a form of decision tree more suited for the special structure of the problem at hand. In this representation, which we call a *deterministic system tree*, there are only two types of nodes: decision nodes and terminal nodes. All branches end with a terminal node, and all terminal nodes appear at the end of branches. Every arc corresponds to a decision, and each arc has an associated gamble. The deterministic system tree for Fig. 1 is shown in Fig. 3. It must be emphasised that, although gambles are acquired upon choosing a decision arc, their value is not discovered until the terminal node is reached. Therefore there is no learning or conditioning involved in this model.

This tree is clearly much more similar to the description of the system. Indeed, normal form decisions for deterministic system trees are again just paths through the tree. How do we find the optimal paths? Following [4], we will use *choice functions on gambles*. Such choice function returns, for every set of gambles, a subset of gambles which are deemed optimal in some sense (which depends on your choice of choice function). For example, maximizing expected utility is one such choice function, but many more exist.

Now, as we saw, each path through the tree has a corresponding gamble. Whence, given a choice function, we can say that a path is optimal whenever its gamble is optimal in the set of gambles induced by all paths. Effectively, we end up with a set of optimal paths.

Two questions arise from this form of solution. The first, addressed by de Cooman and Troffaes, is whether backward induction (more commonly called *dynamic programming* in this field, following Bellman [2]) can be used to reach the normal form solution for a given choice function.

The idea of backward induction is simple. We informally illustrate it on Fig. 3. First, we find which of W, V, and U are optimal. Suppose this is $\{V, W\}$. Then, we determine which of $X+Z$, $Y+W$, and $Y+V$ is optimal. We end up with the backward induction solution, say for instance $\{X+Z, Y+V\}$.

Backward induction thus returns a set of paths, but for many choice functions it can give a different set of paths from the standard normal form solution. In other words, applying the choice function recursively stage-by-stage may not give the same result as applying the choice function on all gambles at once. De Cooman and Troffaes [4] show that backward induction works if the choice function satisfy Bellman's principle of optimality [2] and another property, insensitivity to the omission of non-optimal elements. This paper contributes a reformulation of these results into a theorem about trees and paths in the same fashion as our method for decision trees [6], a proof of necessity as well as sufficiency, and a decomposition of the principle of optimality into two more basic properties.

The second question is whether the normal form solution is equivalent to the combination of local solutions. For instance, in Fig. 3, if W and V are both optimal at N_3, then both $Y+W$ and $Y+V$ should be optimal at N_1, or neither should—this was violated in our earlier example demonstrating backward induction.

This property has been studied extensively for problems modelled by standard decision trees (see for instance [5, 10, 11, 6]). We call a solution with such a property *subtree perfect* (following Selten's analogous concept of subgame perfectness [15]). We show that subtree perfectness for deterministic system trees corresponds to a stricter version of Bellman's principle of optimality obtained by strengthening set inclusions of all properties involved to equalities.

The paper is structured as follow. Section 2 introduces necessary notation. Section 3 presents the results on dynamic programming. Section 4 presents the results on subtree perfectness. Section 5 provides a brief summary of the consequences of the results for the theory of coherent lower previsions. Section 6 concludes the paper.

2 Definitions and Notation

Let Ω be a *possibility space*, i.e. the set of all possible states of nature. Elements ω of Ω are called outcomes. Let $\mathcal{R}$ be a set of rewards (results the subject can receive; they do not have to be desirable rewards). We assume a binary operator $+$ on $\mathcal{R}$, which we call *addition*.[1] We assume that $\mathcal{R}$ has a left identity element 0, so $0+r=r$ for all $r \in \mathcal{R}$. We also assume that $r_1+r_2=r_1+r_3$ implies $r_2=r_3$; this holds for instance if every reward $r \in \mathcal{R}$ has a left inverse $-r \in \mathcal{R}$, so $(-r)+r$ equals the left identity element 0. No other assumptions about $\mathcal{R}$ are required.

A *gamble* is a function $X\colon \Omega \to \mathcal{R}$, with the interpretation that, should ω be the true state of nature, the gamble X gives the subject the reward $X(\omega)$. Addition of gambles is defined in the obvious way: $(X+Y)(\omega)=X(\omega)+Y(\omega)$.

[1] If $\mathcal{R}=\mathbb{R}$, then the operator $+$ does not need to have any resemblance with the usual addition of real numbers, although it is a convenient and popular choice.

Given a set of gambles $\mathcal{X}$ (in this paper, all sets are assumed to be finite, and non-empty unless otherwise noted) from which our subject must pick one, how should she decide? Ideally, she would like to select a single optimal gamble for every set $\mathcal{X}$, but this may not always be possible, for instance, because she lacks information about ω, or because she has no precise utility over her rewards. She might, at least, be able to specify a (possibly empty) set of gambles in $\mathcal{X}$ she considers unacceptable. Any gamble not so judged remains a plausible candidate, and these could be reported as an optimal set. This procedure is represented by a *choice function*: a function that maps sets of options to non-empty subsets.

Definition 1. *A choice function on gambles,* opt*, is a function that maps each set $\mathcal{X}$ of gambles to a non-empty subset of that set:*

$$\emptyset \neq \operatorname{opt}(\mathcal{X}) \subseteq \mathcal{X}.$$

How do we use the concepts of gambles and choice functions to solve deterministic system trees? First, we introduce the concept of normal form decisions, solutions, and operators.

Definition 2. *A normal form decision of a deterministic system tree T is a path through T.*

Definition 3. *The set of all normal form decisions for a deterministic system tree T is denoted by* $\operatorname{nfd}(T)$*.*

Definition 4. *A normal form solution of a deterministic system tree T is a non-empty subset of* $\operatorname{nfd}(T)$*.*

The interpretation of a normal form solution is that the subject may pick any path in this subset and follow it.

Definition 5. *A normal form operator* norm *is a function that maps each deterministic system tree T to a normal form solution of T:*

$$\emptyset \neq \operatorname{norm}(T) \subseteq \operatorname{nfd}(T).$$

Using these definitions, we can define the set of all gambles associated with a deterministic system tree. Recall that any path through a tree has its own gamble, so given the set of all normal form decisions we can find the set of all gambles for that tree.

Definition 6. *The function* gamb *maps deterministic system trees to their set of associated gambles (called* normal form gambles*):*

$$\operatorname{gamb}(T) = \bigcup_{U \in \operatorname{nfd}(T)} \operatorname{gamb}(U).$$

This gives us a set of gambles to which to apply the choice function opt. The procedure is as follows: find the set of normal form gambles, apply the choice function to find an optimal subset of normal form gambles, and then list all normal form decisions with gambles in this optimal subset. This defines a normal form operator, $\mathrm{norm}_{\mathrm{opt}}$.

Definition 7. *For a choice function* opt*, the* normal form operator induced by opt *is defined for any deterministic system tree T by*

$$\mathrm{norm}_{\mathrm{opt}}(T) = \{U \in \mathrm{nfd}(T) \colon \\ \mathrm{gamb}(U) \subseteq \mathrm{opt}(\mathrm{gamb}(T))\}.$$

Of course, since U is always a normal form decision, $\mathrm{gamb}(U)$ is always a singleton in this definition. In particular, the following equality holds:

$$\mathrm{gamb}(\mathrm{norm}_{\mathrm{opt}}(T)) = \mathrm{opt}(\mathrm{gamb}(T)).$$

In the above equation, we have used the following notation: for any set of deterministic system trees $\mathcal{T}$,

$$\mathrm{gamb}(\mathcal{T}) = \bigcup_{T\in\mathcal{T}} \mathrm{gamb}(T).$$

To express backward induction in terms of trees, and to help with many proofs, we introduce a notation for representing a deterministic system tree as a combination of smaller deterministic system trees. For any trees $T_1, \dots, T_n$, we can join them at a decision node, with the arc from this decision node to T_i corresponding to a gamble X_i, and write this as

$$\bigsqcup_{i=1}^{n} X_i T_i.$$

Sometimes we need to work with all the possible ways to join sets of trees $\mathcal{T}_1, \dots, \mathcal{T}_n$ in a similar way. This is written as

$$\bigsqcup_{i=1}^{n} X_i \mathcal{T}_i = \left\{ \bigsqcup_{i=1}^{n} X_i T_i \colon T_i \in \mathcal{T}_i \right\}.$$

This allows gamb to be defined recursively:

$$\mathrm{gamb}\left(\bigsqcup_{i=1}^{n} X_i T_i\right) = \bigcup_{i=1}^{n} (X_i + \mathrm{gamb}(T_i)),$$

where we use the notation

$$X + \mathcal{Y} = \{X + Y \colon Y \in \mathcal{Y}\}.$$

Similarly,

$$\mathrm{gamb}\left(\bigsqcup_{i=1}^{n} X_i \mathcal{T}_i\right) = \bigcup_{i=1}^{n} (X_i + \mathrm{gamb}(\mathcal{T}_i)).$$

Finally, we sometimes need to restrict deterministic system trees to particular subtrees, obtained by removing everything before a certain node.

Definition 8. *A subtree of a deterministic system tree T obtained by removal of all non-descendants of a particular node N, but retaining N, is called the* subtree of T at N *and is denoted by* $\mathrm{st}_N(T)$.

This extends to sets of trees in the usual way:

$$\mathrm{st}_N(\mathcal{T}) = \{\mathrm{st}_N(T) \colon T \in \mathcal{T} \text{ and } N \text{ in } T\}.$$

Usually, the subtrees we need to use are those whose roots are immediate successors of T. Therefore we define $\mathrm{ch}(T)$ to be the set of immediate successors (i.e. children) of the root node of T.

3 Backward Induction Theorem

We introduce a new normal form operator based on backward induction, defined recursively. The operator works by eliminating non-optimal paths in subtrees, then bringing all optimal paths to the next largest subtree, and so on until the root node is reached. To do so elegantly, we extend $\mathrm{norm}_{\mathrm{opt}}$ to act upon sets of trees:

$$\mathrm{norm}_{\mathrm{opt}}(\mathcal{T}) = \{U \in \mathrm{nfd}(\mathcal{T}) \colon \\ \mathrm{gamb}(U) \subseteq \mathrm{opt}(\mathrm{gamb}(\mathcal{T}))\}.$$

Definition 9. *The normal form operator* $\mathrm{back}_{\mathrm{opt}}$ *is defined for any deterministic system tree T that consists only of a terminal node by*

$$\mathrm{back}_{\mathrm{opt}}(T) = T$$

and for any other deterministic system tree $T = \bigsqcup_{i=1}^{n} X_i T_i$ *by*

$$\mathrm{back}_{\mathrm{opt}}(T) = \mathrm{norm}_{\mathrm{opt}}\left(\bigsqcup_{i=1}^{n} X_i \,\mathrm{back}_{\mathrm{opt}}(T_i)\right).$$

We are interested in determining when $\mathrm{back}_{\mathrm{opt}}$ and $\mathrm{norm}_{\mathrm{opt}}$ coincide. This happens if and only if the following two properties hold.

Property 1 (Insensitivity of optimality to the omission of non-optimal elements)**.** *For any sets of gambles $\mathcal{X}$ and $\mathcal{Y}$,*

$$\mathrm{opt}(\mathcal{X}) \subseteq \mathcal{Y} \subseteq \mathcal{X} \Rightarrow \mathrm{opt}(\mathcal{Y}) = \mathrm{opt}(\mathcal{X}).$$

De Cooman and Troffaes [4] explain that this property is crucial for backward induction to work. It also appears in the work of Sen [16], who shows it to be one "half" of the property of *path independence* (see for instance Plott [12]).

The second property was introduced by Bellman [2] with the following explanation:

> An optimal policy has the property that, whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.

Note that, in the context of deterministic system trees, states are simply decision nodes.

Although Bellman states the principle in terms of the first decision only, it implies that the restriction of an optimal policy to any subtree must be optimal. We formalize the principle into the following property.

Property 2 (Principle of Optimality). *A normal form operator* norm *satisfies the principle of optimality if, for any deterministic system tree T, and any node N in at least one element of* $\operatorname{norm}(T)$*,*

$$\operatorname{st}_N(\operatorname{norm}(T)) \subseteq \operatorname{norm}(\operatorname{st}_N(T)).$$

Equivalently, for any normal form decision $U \in$ $\operatorname{norm}(T)$ *and any node N in U,*

$$\operatorname{st}_N(U) \in \operatorname{norm}(\operatorname{st}_N(T)).$$

For the particular case of $\operatorname{norm}_{\mathrm{opt}}$, the above definition is easily seen to be equivalent to the inclusion formula of de Cooman and Troffaes [4, Definition 13]—but our notation is far more efficient at expressing it.

Interestingly, we can decompose Property 2, the principle of optimality, into two far more basic properties.

Property 3 (Preservation of non-optimality under the addition of elements). *For any sets of gambles $\mathcal{X}$ and $\mathcal{Y}$,*

$$\mathcal{Y} \subseteq \mathcal{X} \Rightarrow \operatorname{opt}(\mathcal{Y}) \supseteq \operatorname{opt}(\mathcal{X}) \cap \mathcal{Y}.$$

This is a type of independence of irrelevant alternatives (see [1, 13]), called property α by Sen [16]. It is the other "half" of path independence (so we show that path independence is necessary for dynamic programming). Property 3 is not explicitly invoked by de Cooman and Troffaes, but it is used in a proof for a particular choice function [4, Proposition 16].

Property 4 (Backward Addition Property). *For any gamble X and any non-empty finite set of gambles $\mathcal{Y}$,*

$$\operatorname{opt}(X + \mathcal{Y}) \subseteq X + \operatorname{opt}(\mathcal{Y}).$$

This property was informally foreseen by de Cooman and Troffaes (see the discussion of "additivity" [4, §3.4]). It is similar to properties relating to backward induction for other decision processes [6, 17].

The proof of equivalence relies on the next lemma.

Lemma 10. *Let* norm *be any normal form operator. Let T be a consistent decision tree. If,*

(i) for all nodes $K \in \operatorname{ch}(T)$ such that K is in at least one element of $\operatorname{norm}(T)$*,*

$$\operatorname{st}_K(\operatorname{norm}(T)) \subseteq \operatorname{norm}(\operatorname{st}_K(T)),$$

(ii) and, for all nodes $K \in \operatorname{ch}(T)$, and all nodes $L \in$ $\operatorname{st}_K(T)$ *such that L is in at least one element of* $\operatorname{norm}(\operatorname{st}_K(T))$*,*

$$\operatorname{st}_L(\operatorname{norm}(\operatorname{st}_K(T))) \subseteq \operatorname{norm}(\operatorname{st}_L(\operatorname{st}_K(T))),$$

then, for all nodes N in T such that N is in at least one element of $\operatorname{norm}(T)$*,*

$$\operatorname{st}_N(\operatorname{norm}(T)) \subseteq \operatorname{norm}(\operatorname{st}_N(T)).$$

Proof. If N is the root of T, then the result is immediate. If $N \in \operatorname{ch}(T)$, then the result follows from (i). Otherwise, N must be in $\operatorname{st}_K(T)$ for one $K \in \operatorname{ch}(T)$.

By assumption, there is a $U \in \operatorname{norm}(T)$ that contains N (and of course also K). Therefore, $U \in \operatorname{st}_K(\operatorname{norm}(T))$, and by (i), $\operatorname{st}_K(U) \in \operatorname{norm}(\operatorname{st}_K(T))$, and so N is also in at least one element of $\operatorname{norm}(\operatorname{st}_K(T))$.

We use the fact that, if $\mathcal{U}$ and $\mathcal{V}$ are sets of normal form decisions such that $\mathcal{U} \subseteq \mathcal{V}$, then for any node N, $\operatorname{st}_N(\mathcal{U}) \subseteq \operatorname{st}_N(\mathcal{V})$. Combining everything, by (i),

$$\operatorname{st}_N(\operatorname{st}_K(\operatorname{norm}(T))) \subseteq \operatorname{st}_N(\operatorname{norm}(\operatorname{st}_K(T)))$$

hence, since N is in at least one element of $\operatorname{norm}(\operatorname{st}_K(T))$, by (ii) we have

$$\subseteq \operatorname{norm}(\operatorname{st}_N(\operatorname{st}_K(T))),$$

whence the desired result follows, since $\operatorname{st}_N(\operatorname{st}_K(T)) = \operatorname{st}_N(T)$. □

Theorem 11. $\operatorname{norm}_{\mathrm{opt}}$ *satisfies Property 2 if and only if* opt *satisfies Properties 3 and 4.*

Proof. "only if". Let X be a gamble and $\mathcal{Y} = \{Y_1, \dots, Y_n\}$ be a set of gambles. Consider the upper tree in Fig. 4. If $X + Y_k \in \operatorname{opt}(X + \mathcal{Y})$, then by Property 2 it follows that $Y \in \operatorname{opt}(\mathcal{Y})$, hence Property 4 holds. Next, consider the lower tree. Let $\mathcal{Y} = \{Y_1 \dots, Y_m\}$, $\mathcal{Z} = \{Z_1, \dots, Z_n\}$ and suppose

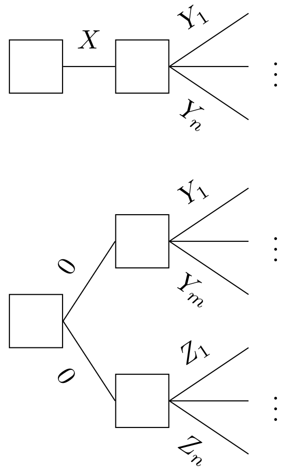

Figure 4: Decision trees for Theorem 11.

$\mathcal{Y} \cap \mathcal{Z} = \emptyset$. Now let $\mathcal{X} = \mathcal{Y} \cup \mathcal{Z}$. By Property 2 we know that if $Y \in \mathcal{Y} \cap \operatorname{opt}(\mathcal{X})$, then $Y \in \operatorname{opt}(\mathcal{Y})$, hence Property 3 holds.

"if". We proceed by structural induction. Let T be a deterministic system tree. The base step, to show the result when T consists of a terminal node only, is trivial. The inductive step is to suppose that Property 2 holds for every $\operatorname{st}_K(T)$ where $K \in \operatorname{ch}(T)$, and then show that Property 2 holds for T. By Lemma 10, we need only show that for every $K \in \operatorname{ch}(T)$ that is in at least one element of $\operatorname{norm}_{\mathrm{opt}}(T)$,

$$\operatorname{st}_K(\operatorname{norm}_{\mathrm{opt}}(T)) \subseteq \operatorname{norm}_{\mathrm{opt}}(\operatorname{st}_K(T)).$$

So, the proof is established if we can show that, for every $U \in \operatorname{norm}_{\mathrm{opt}}(T)$ passing through $K \in \operatorname{ch}(T)$,

$$\operatorname{st}_K(U) \in \operatorname{norm}_{\mathrm{opt}}(\operatorname{st}_K(T)). \tag{1}$$

We now express this in terms of gambles—but first we introduce some notation.

Let $\operatorname{ch}(T) = \{K_1, \dots, K_n\}$, and $K = K_k$. Let $\operatorname{gamb}(\operatorname{st}_{K_i}(T)) = \mathcal{Y}_i$, and let X_i be the gamble corresponding to the arc to K_i. That is,

$$T = \bigsqcup_{i=1}^{n} X_i \operatorname{st}_{K_i}(T).$$

Recall, U contains the node K_k, so $\operatorname{gamb}(U) = X_k + Y_k$ for some $Y_k \in \mathcal{Y}_k$.

Now, because $U \in \operatorname{norm}_{\mathrm{opt}}(T)$, we know that

$$X_k + Y_k \in \operatorname{opt}(\operatorname{gamb}(T)) = \operatorname{opt}\left(\bigcup_{i=1}^{n}(X_i + \mathcal{Y}_i)\right). \tag{2}$$

To establish Eq. (1), we must simply show that $Y_k \in \operatorname{opt}(\mathcal{Y}_k)$.

Indeed. Obviously,

$$X_k + \mathcal{Y}_k \subseteq \bigcup_{i=1}^{n}(X_i + \mathcal{Y}_i).$$

Applying Property 3,

$$\operatorname{opt}(X_k + \mathcal{Y}_k) \supseteq \operatorname{opt}\left(\bigcup_{i=1}^{n}(X_i + \mathcal{Y}_i)\right) \cap (X_k + \mathcal{Y}_k).$$

However, by Eq. (2), $X_k + Y_k$ belongs to the right hand side, whence, it must also belong to the left hand side. Now, apply Property 4, to see that indeed $Y_k \in \operatorname{opt}(\mathcal{Y}_k)$. This completes the inductive step. □

We are now in a position to prove a backward induction theorem. It turns out that we can incorporate another simple concept into this theorem, namely that of *strategic equivalence*. Two trees are strategically equivalent if their set of gambles is the same. We can show easily that $\operatorname{back}_{\mathrm{opt}}$ and $\operatorname{norm}_{\mathrm{opt}}$ agreeing is equivalent to $\operatorname{back}_{\mathrm{opt}}$ preserving strategic equivalence.

Theorem 12. *Let* opt *be any choice function. The following conditions are equivalent.*

(A) For any deterministic system tree T, it holds that $\operatorname{back}_{\mathrm{opt}}(T) = \operatorname{norm}_{\mathrm{opt}}(T)$.

(B) For any strategically equivalent deterministic system trees, T_1 and T_2, it holds that

$$\operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_1)) = \operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_2)).$$

(C) opt *satisfies Properties 1 and 2.*

Lemma 13. *If, for all strategically equivalent deterministic system trees T_1 and T_2, it holds that*

$$\operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_1)) = \operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_2)),$$

then opt *satisfies Property 1.*

Proof. Let $\mathcal{X}$ and $\mathcal{Y} = \{Y_1, \dots, Y_n\}$ be sets of gambles such that $\operatorname{opt}(\mathcal{X}) \subseteq \mathcal{Y} \subseteq \mathcal{X}$. Let T_1 be a deterministic system tree with just one decision node and $\operatorname{gamb}(T_1) = \mathcal{X}$. Let T_2 be a deterministic system tree constructed as follows: there is one decision arc with gamble 0 that leads to T_1, and n other decision arcs, each leading immediately to a terminal node, with gambles Y_1 to Y_n. Clearly, $\operatorname{gamb}(T_2) = \mathcal{X}$. We have

$$\operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_2)) = \operatorname{opt}(\operatorname{opt}(\mathcal{X}) \cup \mathcal{Y}) = \operatorname{opt}(\mathcal{Y}).$$

because $\operatorname{opt}(\mathcal{X}) \subseteq \mathcal{Y}$. Since $\operatorname{back}_{\mathrm{opt}}$ is assumed to preserve strategic equivalence, and T_1 and T_2 are strategically equivalent by construction, it follows that $\operatorname{opt}(\mathcal{Y}) = \operatorname{opt}(\mathcal{X})$, as required. □

Lemma 14. *If, for all strategically equivalent deterministic system trees T_1 and T_2, it holds that*

$$\operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_1)) = \operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_2)),$$

then $\operatorname{norm}_{\mathrm{opt}}$ *satisfies Property 2.*

Proof. We show that opt must satisfy Properties 3 and 4 and invoke Theorem 11. We can again use the two trees from Fig. 4. Let the upper tree be called T_1, and let T_2 be a tree with only one decision node and $\operatorname{gamb}(T_2) = X + \mathcal{Y}$. Then,

$$\begin{aligned}\operatorname{opt}(X+\mathcal{Y}) &= \operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_2))\\ &= \operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_1))\\ &= \operatorname{opt}(X+\operatorname{opt}(\mathcal{Y})) \subseteq X + \operatorname{opt}(\mathcal{Y}),\end{aligned}$$

so Property 4 holds.

Let T_1 be the lower tree in Fig. 4, with $\{\mathcal{Y},\mathcal{Z}\}$ a partition of $\mathcal{X}$. Let T_2 have one decision node and $\operatorname{gamb}(T_2) = \mathcal{X}$. As assumed, $\operatorname{gamb}(\operatorname{back}_{\mathrm{opt}}(T_1)) = \operatorname{opt}(\operatorname{opt}(\mathcal{Y}) \cup \operatorname{opt}(\mathcal{Z})) = \operatorname{opt}(\mathcal{X})$. So,

$$\begin{aligned}\operatorname{opt}(\mathcal{X}) \cap \mathcal{Y} &= \operatorname{opt}(\operatorname{opt}(\mathcal{Y}) \cup \operatorname{opt}(\mathcal{Z})) \cap \mathcal{Y}\\ &\subseteq (\operatorname{opt}(\mathcal{Y}) \cup \operatorname{opt}(\mathcal{Z})) \cap \mathcal{Y}\\ &= \operatorname{opt}(\mathcal{Y}) \cap \mathcal{Y} = \operatorname{opt}(\mathcal{Y}),\end{aligned}$$

so Property 3 holds. □

Lemma 15. *If $\mathcal{T} \subseteq \mathcal{U} \subseteq \mathcal{V}$ are sets of deterministic system trees,* opt *satisfies Property 1, and* $\operatorname{norm}_{\mathrm{opt}}(\mathcal{T}) = \operatorname{norm}_{\mathrm{opt}}(\mathcal{V})$*, then* $\operatorname{norm}_{\mathrm{opt}}(\mathcal{U}) = \operatorname{norm}_{\mathrm{opt}}(\mathcal{V})$*.*

Proof. By assumption, we have that

$$\begin{aligned}\operatorname{opt}(\operatorname{gamb}(\mathcal{V})) = \operatorname{opt}(\operatorname{gamb}(\mathcal{T})) &\subseteq \operatorname{gamb}(\mathcal{T})\\ &\subseteq \operatorname{gamb}(\mathcal{U}) \subseteq \operatorname{gamb}(\mathcal{V}).\end{aligned}$$

Hence, by Property 1,

$$\operatorname{opt}(\operatorname{gamb}(\mathcal{T})) = \operatorname{opt}(\operatorname{gamb}(\mathcal{U})) = \operatorname{opt}(\operatorname{gamb}(\mathcal{V})).$$

So,

$$\begin{aligned}\operatorname{norm}_{\mathrm{opt}}(\mathcal{U}) &= \{U \in \mathcal{U}\colon\ \operatorname{gamb}(U) \subseteq \operatorname{opt}(\operatorname{gamb}(\mathcal{T}))\}\\ &\supseteq \{U \in \mathcal{T}\colon\ \operatorname{gamb}(U) \subseteq \operatorname{opt}(\operatorname{gamb}(\mathcal{T}))\}\\ &= \operatorname{norm}_{\mathrm{opt}}(\mathcal{T})\end{aligned}$$

because $\mathcal{U} \supseteq \mathcal{T}$, and

$$\begin{aligned}\operatorname{norm}_{\mathrm{opt}}(\mathcal{U}) &= \{U \in \mathcal{U}\colon\ \operatorname{gamb}(U) \subseteq \operatorname{opt}(\operatorname{gamb}(\mathcal{V}))\}\\ &\subseteq \{U \in \mathcal{V}\colon\ \operatorname{gamb}(U) \subseteq \operatorname{opt}(\operatorname{gamb}(\mathcal{V}))\}\\ &= \operatorname{norm}_{\mathrm{opt}}(\mathcal{V})\end{aligned}$$

because $\mathcal{U} \subseteq \mathcal{V}$. We conclude that

$$\operatorname{norm}_{\mathrm{opt}}(\mathcal{T}) \subseteq \operatorname{norm}_{\mathrm{opt}}(\mathcal{U}) \subseteq \operatorname{norm}_{\mathrm{opt}}(\mathcal{V}).$$

Now use $\operatorname{norm}_{\mathrm{opt}}(\mathcal{T}) = \operatorname{norm}_{\mathrm{opt}}(\mathcal{V})$. □

Proof of Theorem 12. (A) $\implies$ (B). Immediate, since for strategically equivalent trees, $\operatorname{norm}_{\mathrm{opt}}(T_1) = \operatorname{norm}_{\mathrm{opt}}(T_2)$ by definition.

(B) $\implies$ (C). See Lemmas 13 and 14.

(C) $\implies$ (A). We proceed by structural induction. The base step is trivial. The induction hypothesis is that, for a $T = \bigsqcup_{i=1}^{n} X_i T_i$, we have $\operatorname{norm}_{\mathrm{opt}}(T_i) = \operatorname{back}_{\mathrm{opt}}(T_i)$ for all i. The induction step is to show that this implies $\operatorname{norm}_{\mathrm{opt}}(T) = \operatorname{back}_{\mathrm{opt}}(T)$.

Let K_i be the root node of T_i. For any i such that K_i is in at least one element of $\operatorname{norm}_{\mathrm{opt}}(T)$, we know from Property 2 that $\operatorname{st}_{K_i}(\operatorname{norm}_{\mathrm{opt}}(T)) \subseteq \operatorname{norm}_{\mathrm{opt}}(T_i) = \operatorname{back}_{\mathrm{opt}}(T_i)$. If instead K_i is not in at least one element of $\operatorname{norm}_{\mathrm{opt}}(T)$, then nothing from $\operatorname{back}_{\mathrm{opt}}(T_i)$ is involved in $\operatorname{norm}_{\mathrm{opt}}(T)$. Therefore,

$$\operatorname{norm}_{\mathrm{opt}}(T) \subseteq \bigsqcup_{i=1}^{n} X_i \operatorname{back}_{\mathrm{opt}}(T_i) \subseteq \operatorname{nfd}(T).$$

Since $\operatorname{norm}_{\mathrm{opt}}(\operatorname{nfd}(T)) = \operatorname{norm}_{\mathrm{opt}}(T)$ and it follows from Property 1 that $\operatorname{norm}_{\mathrm{opt}}(\operatorname{norm}_{\mathrm{opt}}(T)) = \operatorname{norm}_{\mathrm{opt}}(T)$,[2] we can use Lemma 15 to conclude that

$$\begin{aligned}\operatorname{back}_{\mathrm{opt}}(T) &= \operatorname{norm}_{\mathrm{opt}}\left(\bigsqcup_{i=1}^{n} X_i \operatorname{back}_{\mathrm{opt}}(T_i)\right)\\ &= \operatorname{norm}_{\mathrm{opt}}(T).\end{aligned}$$

□

4 Subtree Perfectness

Subtree perfectness means that, when a normal form solution is restricted to a subtree of a deterministic system tree, it is equal to the solution of the subtree.

Definition 16. *A normal form operator* norm *is subtree perfect if, for any deterministic system tree T, and any node N in at least one element of* $\operatorname{norm}(T)$*,*

$$\operatorname{st}_N(\operatorname{norm}(T)) = \operatorname{norm}(\operatorname{st}_N(T)).$$

This is just a stronger form of Property 2, and so it is unsurprising that the necessary and sufficient conditions on opt turn out to be identical apart from having equalities instead of inclusions.

Property 5 (Intersection property)**.** *For any sets of gambles $\mathcal{X}$ and $\mathcal{Y}$ such that $\mathcal{Y} \subseteq \mathcal{X}$ and $\operatorname{opt}(\mathcal{X}) \cap \mathcal{Y} \neq \emptyset$,*

$$\operatorname{opt}(\mathcal{Y}) = \operatorname{opt}(\mathcal{X}) \cap \mathcal{Y}.$$

Property 6 (Addition Property)**.** *For any gamble X and any non-empty finite set of gambles $\mathcal{Y}$,*

$$\operatorname{opt}(X + \mathcal{Y}) = X + \operatorname{opt}(\mathcal{Y}).$$

[2] Use $\mathcal{Y} = \operatorname{opt}(\mathcal{X})$ in Property 1.

Note that Property 1 is actually included within Property 5 (which is in fact equivalent to saying that opt defines a total preorder [1]). Another useful reformulation of Property 5 is [16, 6]:

Property 7 (Very strong path independence)**.** *For any sets of gambles $\mathcal{X}_1$, …,$\mathcal{X}_n$, let $\mathcal{I} = \{i\colon \mathcal{X}_i \cap \mathrm{opt}(\cup_{i=1}^n \mathcal{X}_i) \neq \emptyset\}$. Then,*

$$\mathrm{opt}\left(\bigcup_{i=1}^n \mathcal{X}_i\right) = \bigcup_{i\in\mathcal{I}}^n \mathrm{opt}(\mathcal{X}_i).$$

Theorem 17. *The normal form operator* $\mathrm{norm}_{\mathrm{opt}}$ *is subtree perfect for deterministic system trees if and only if* opt *satisfies Properties 5 and 6.*

Lemma 18. *Consider a deterministic system tree $T = \bigsqcup_{i=1}^n X_i T_i$, and any choice function* opt. *For each tree T_i, let K_i be its root. Then, K_i is in at least one element of* $\mathrm{norm}_{\mathrm{opt}}(T)$ *if and only if*

$$(X_i + \mathrm{gamb}(T_i)) \cap \mathrm{opt}(\mathrm{gamb}(T)) \neq \emptyset. \tag{3}$$

Proof. Eq. (3) holds if and only if there is a normal form decision $U \in \mathrm{nfd}(T_i)$ such that $X_i + \mathrm{gamb}(U) \subseteq \mathrm{opt}(\mathrm{gamb}(T))$. This is equivalent to there being a U such that $\mathrm{gamb}(\sqcup X_i U) \subseteq \mathrm{opt}(\mathrm{gamb}(T))$. Clearly, $\sqcup X_i U$ is a normal form decision of T, and so by definition of $\mathrm{norm}_{\mathrm{opt}}$, Eq. (3) holds if and only if $\sqcup X_i U$ is in $\mathrm{norm}_{\mathrm{opt}}(T)$, which holds if and only if K_i is in at least one element of $\mathrm{norm}_{\mathrm{opt}}(T)$. □

Lemma 19. *If $T = \bigsqcup_{i=1}^n X_i T_i$, and* opt *is a choice function satisfying Properties 5 and 6, then*

$$\mathrm{gamb}(\mathrm{norm}_{\mathrm{opt}}(T)) = \bigcup_{i\in\mathcal{I}} (X_i + \mathrm{gamb}(\mathrm{norm}_{\mathrm{opt}}(T_i))) \tag{4}$$

implies

$$\mathrm{norm}_{\mathrm{opt}}(T) = \mathrm{nfd}\left(\bigsqcup_{i\in\mathcal{I}} X_i\, \mathrm{norm}_{\mathrm{opt}}(T_i)\right),$$

where $\mathcal{I} = \{i \in \{1,\dots,n\}\colon (X_i + \mathrm{gamb}(T_i)) \cap \mathrm{opt}(\mathrm{gamb}(T)) \neq \emptyset\}$.

Proof. We first show that

$$\mathrm{norm}_{\mathrm{opt}}(T) \supseteq \mathrm{nfd}\left(\bigsqcup_{i\in\mathcal{I}} X_i\, \mathrm{norm}_{\mathrm{opt}}(T_i)\right).$$

Consider a normal form decision $U \in \mathrm{nfd}\left(\bigsqcup_{i\in\mathcal{I}} X_i\, \mathrm{norm}_{\mathrm{opt}}(T_i)\right)$. To show that $U \in \mathrm{norm}_{\mathrm{opt}}(T)$, we must show that $U \in \mathrm{nfd}(T)$ and $\mathrm{gamb}(U) \subseteq \mathrm{gamb}(\mathrm{norm}_{\mathrm{opt}}(T))$. The former is obvious, and the latter is established by Eq. (4):

$$\begin{aligned}\mathrm{gamb}(U) &\subseteq \bigcup_{i\in\mathcal{I}} (X_i + \mathrm{gamb}(\mathrm{norm}_{\mathrm{opt}}(T_i))) \\ &= \mathrm{gamb}(\mathrm{norm}_{\mathrm{opt}}(T)).\end{aligned}$$

Next we show that

$$\mathrm{norm}_{\mathrm{opt}}(T) \subseteq \mathrm{nfd}\left(\bigsqcup_{i\in\mathcal{I}} X_i\, \mathrm{norm}_{\mathrm{opt}}(T_i)\right).$$

Let $U \in \mathrm{norm}_{\mathrm{opt}}(T)$. Let V be U with the root node removed, that is, $U = \sqcup X_k V$ for some k. Clearly, $V \in \mathrm{nfd}(T_k)$. It suffices to show that $V \in \mathrm{norm}_{\mathrm{opt}}(T_k)$. Let $\{Y\} = \mathrm{gamb}(V)$ and let $\mathcal{Y} = \mathrm{gamb}(T_k)$. We know that $X_k + Y \in \mathrm{gamb}(T)$, and $Y \in \mathrm{gamb}(T_k)$. Also, $X_k + \mathcal{Y} \subseteq \mathrm{gamb}(T)$. By Property 5 and Lemma 18,

$$\mathrm{opt}(X_k + \mathcal{Y}) = \mathrm{opt}(\mathrm{gamb}(T)) \cap (X_k + \mathcal{Y}).$$

By Property 6,

$$X_k + \mathrm{opt}(\mathcal{Y}) = \mathrm{opt}(X_k + \mathcal{Y}),$$

whence

$$X_k + \mathrm{opt}(\mathcal{Y}) = \mathrm{opt}(\mathrm{gamb}(T)) \cap (X_k + \mathcal{Y}).$$

We know $X_k + Y$ is in the right hand side, so $X_k + Y$ is in the left hand side. Therefore $Y \in \mathrm{opt}(\mathcal{Y})$ and $V \in \mathrm{norm}_{\mathrm{opt}}(T_k)$. □

Lemma 20 (Huntley and Troffaes [6, Lemma 17])**.** *Let* norm *be a normal form operator. Let T be a deterministic system tree. If,*

(i) for all nodes $K \in \mathrm{ch}(T)$ such that K is in at least one element of $\mathrm{norm}(T)$*,*

$$\mathrm{st}_K(\mathrm{norm}(T)) = \mathrm{norm}(\mathrm{st}_K(T)),$$

(ii) and, for all nodes $K \in \mathrm{ch}(T)$, and all nodes $L \in \mathrm{st}_K(T)$ such that L is in at least one element of $\mathrm{norm}(\mathrm{st}_K(T))$*,*

$$\mathrm{st}_L(\mathrm{norm}(\mathrm{st}_K(T))) = \mathrm{norm}(\mathrm{st}_L(\mathrm{st}_K(T))),$$

then, for all nodes N in T such that N is in at least one element of $\mathrm{norm}(T)$*,*

$$\mathrm{st}_N(\mathrm{norm}(T)) = \mathrm{norm}(\mathrm{st}_N(T)).$$

Lemma 21. *If* $\mathrm{norm}_{\mathrm{opt}}$ *is subtree perfect then* opt *satisfies Property 5.*

Proof. Let $\mathcal{X}$ and $\mathcal{Y}$ be sets of gambles such that $\mathcal{Y} \subseteq \mathcal{X}$. Let T_1 and T_2 be deterministic system trees with exactly one decision node, and $\mathrm{gamb}(T_1) = \mathcal{X}$, $\mathrm{gamb}(T_2) = \mathcal{Y}$. Let $T = T_1 \sqcup T_2$ (so the arcs to T_1 and T_2 have reward 0), and N be the node at the root of T_2. So, $\mathrm{gamb}(T) = \mathcal{X}$. Now, $\mathrm{gamb}(\mathrm{norm}_{\mathrm{opt}}(T)) = \mathrm{opt}(\mathcal{X})$, and $\mathrm{gamb}(\mathrm{st}_N(\mathrm{norm}_{\mathrm{opt}}(T))) = \mathrm{gamb}(\mathcal{Y}) \cap \mathrm{opt}(\mathcal{X})$. By subtree perfectness, Property 5 follows. □

Lemma 22. *If* norm_{opt} *is subtree perfect, then* opt *satisfies Property 6.*

Proof. Let X be a gamble and let $\mathcal{Y}$ be a non-empty finite set of gambles. Let T_1 be a deterministic system tree with exactly one decision node and $\text{gamb}(T_1) = \mathcal{Y}$. Let $T = \sqcup X T_1$, so $\text{gamb}(T) = X + \mathcal{Y}$. Now,

$$\text{gamb}(\text{norm}_{\text{opt}}(T)) = \text{opt}(X + \mathcal{Y})$$

and

$$\text{gamb}(\text{norm}_{\text{opt}}(T_1)) = \text{opt}(\mathcal{Y}).$$

By subtree perfectness and the definition of norm_{opt}, we must have that, first, any gamble $X + Y \in \text{opt}(X + \mathcal{Y})$ must have $Y \in \text{opt}(\mathcal{Y})$ (else there is a $U \in \text{norm}_{\text{opt}}(T)$ that is non-optimal in T_1), and second, any $Y \in \text{opt}(\mathcal{Y})$ must have $X + Y \in \text{opt}(X + \mathcal{Y})$ (else there is a $U \in \text{norm}_{\text{opt}}(T_1)$ with $\sqcup X U$ non-optimal in T). Therefore $\text{opt}(X + \mathcal{Y}) = X + \text{opt}(\mathcal{Y})$. □

Proof of Theorem 17. "only if". Follows from Lemmas 21 and 22.

"if". We proceed by structural induction as usual. The base step is trivial. The induction hypothesis is that, for a $T = \bigsqcup_{i=1}^{n} X_i T_i$, we have subtree perfectness at all T_i. If we can show that

$$\text{gamb}(\text{norm}_{\text{opt}}(T)) = \bigcup_{i \in \mathcal{I}} (X_i + \text{gamb}(\text{norm}_{\text{opt}}(T_i)))$$

for $\mathcal{I} = \{i \in \{1, \dots, n\} \colon (X_i + \text{gamb}(T_i)) \cap \text{opt}(\text{gamb}(T)) \neq \emptyset\}$, then by Lemma 19 and Lemma 20, subtree perfectness holds for T.

We have

$$\text{gamb}(\text{norm}_{\text{opt}}(T)) = \text{opt}\left(\bigcup_{i=1}^{n} (X_i + \text{gamb}(T_i))\right)$$

whence by Property 7

$$= \bigcup_{i \in \mathcal{I}} \text{opt}(X_i + \text{gamb}(T_i))$$

whence by Property 6

$$= \bigcup_{i \in \mathcal{I}} (X_i + \text{opt}(\text{gamb}(T_i)))$$
$$= \bigcup_{i \in \mathcal{I}} (X_i + \text{gamb}(\text{norm}_{\text{opt}}(T_i)))$$

as required. □

	Property				
	1	3	4	5	6
E-admissibility	✓	✓	✓		✓
Maximality	✓	✓	✓		✓
Γ-maximin	✓	✓		✓	
Interval Dominance	✓	✓			

Table 1: Properties of various choice functions.

5 Imprecise Probability

De Cooman and Troffaes [4, §3.2–3.5] investigate whether dynamic programming works for four common choice functions in *imprecise probability* [18], namely maximality, E-admissibility, Γ-maximin, and interval dominance. The first two satisfy all properties, and the latter two fail Property 4. Γ-maximin and interval dominance fail because of the non-additivity of a coherent lower prevision.

For subtree perfectness, none of the choice functions satisfies all the necessary properties. Property 5 requires a total preorder, and, of the four, only Γ-maximin is. Since Γ-maximin fails Property 4, it automatically fails Property 6. These results mirror those for standard decision trees [6]: only maximality and E-admissibility allow backward induction, and nothing is subtree perfect. A table showing the properties satisfied by each choice function is shown in Table 1.

As mentioned by de Cooman and Troffaes, Γ-maximin could satisfy Property 6 for certain lower previsions. Suppose that Ω is a product of possibility spaces $\Omega_1, \dots, \Omega_m$, and the gambles on the ith decision arc in any path is a gamble on Ω_i. If the overall lower prevision $\underline{P}$ is a suitable independent product of lower previsions $\underline{P}_i$ on the Ω_i, then additivity will be satisfied. We refer to [4, §3.4] for more details and references.

6 Conclusion

In this paper we have investigated dynamic programming for deterministic discrete-time systems with uncertain gain using normal form operators induced by choice functions. We have brought the work of de Cooman and Troffaes into the decision tree setting of [6]. In doing so, we have extended their Bellman Equation Theorem [4, Theorem 14] by adding necessity to their sufficiency, allowing arbitrary rewards (so a utility function over rewards is no longer assumed), and fairly arbitrary addition operators. Also, we have decomposed Bellman's principle of optimality into two much simpler properties.

Further, we have found simple necessary and sufficient conditions for subtree perfectness, which is a stronger form of Bellman's principle. The distinction between dynamic programming and subtree perfectness is not often made (see for instance the informal description of Property 2 by Luenberger [9, p. 419]: this is clearly subtree perfectness being described).

A likely reason for this lack of distinction is that, under the assumption of a total preorder (a very popular assumption in decision theory literature) the two concepts become almost identical. We cannot think of a well-known choice function for any uncertainty model that satisfies Properties 4 and 5 but not Property 6. The distinction is much more important with imprecise methods, where a major attraction is the ability to model indecision and incomparability of options. In such cases, subtree perfectness will always fail.

The key observations are that lack of subtree perfectness is not necessarily a barrier to dynamic programming, but nor is success of dynamic programming enough to guarantee that one's normal form solution is completely well-behaved.

Acknowledgements

The first author is supported by the EPSRC. We thank both reviewers for their valuable comments and suggestions.

References

[1] Kenneth J. Arrow. Rational choice functions and orderings. *Economica*, 26(102):121–127, May 1959.

[2] R. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, 1957.

[3] Robert T. Clemen and Terence Reilly. *Making Hard Decisions.* Duxbury, 2001.

[4] G. De Cooman and M.C.M. Troffaes. Dynamic programming for deterministic discrete-time systems with uncertain gain. *International Journal of Approximate Reasoning*, 39(2-3):257–278, Jun 2005.

[5] P. Hammond. Consequentialist foundations for expected utility. *Theory and Decision*, 25(1):25–78, Jul 1988.

[6] N. Huntley and M. C. M. Troffaes. Characterizing factuality in normal form sequential decision making. In Thomas Augustin, Frank P. A. Coolen, Serafin Moral, and Matthias C. M. Troffaes, editors, *ISIPTA'09: Proceedings of the Sixth International Symposium on Imprecise Probability: Theories and Applications*, pages 239–248, 2009.

[7] D. V. Lindley. *Making Decisions.* Wiley, London, 2nd edition, 1985.

[8] R.D. Luce and H. Raiffa. *Games and Decisions: introduction and critical survery.* Wiley, 1957.

[9] D. G. Luenberger. *Introduction to Dynamic Systems.* Wiley, 1979.

[10] M.J. Machina. Dynamic consistency and non-expected utility models of choice under uncertainty. *Journal of Economic Literature*, 27(1622-1688), 1989.

[11] E. F. McClennen. *Rationality and Dynamic Choice: Foundational Explorations.* Cambridge University Press, 1990.

[12] C.R. Plott. Path independence, rationality, and social choice. *Econometrica*, 41(6):1075–1091, Nov 1973.

[13] P. Ray. Independence of irrelevant alternatives. *Econometrica*, 41(5):987–991, Sep 1973.

[14] Teddy Seidenfeld. When normal and extensive form decisions differ. In D. Prawitz, B. Skyrms, and D. Westerstahl, editors, *Logic, Methodology and Philosophy of Science IX, Proceedings of the Ninth International Congress of Logic, Methodology and Philosophy of Science*, volume 134 of *Studies in Logic and the Foundations of Mathematics*, pages 451–463. Elsevier, 1995.

[15] R. Selten. Reexamination of the perfectness concept for equilibrium points in extensive games. *International Journal of Game Theory*, 4(1):25–55, Mar 1975.

[16] A. K. Sen. Social choice theory: A re-examination. *Econometrica*, 45(1):53–89, 1977.

[17] M. C. M. Troffaes, N. Huntley, and R. Shirota Filho. Sequential decision processes under act-state independence with arbitrary choice functions. In E. Huellermeier, R. Kruse, and F. Hoffmann, editors, *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 98–107. Springer, 2010.

[18] P. Walley. *Statistical Reasoning with Imprecise Probabilities.* Chapman and Hall, London, 1991.