Backward Induction
for Sequential Decision Problems

Nathan Huntley   Matthias C. M. Troffaes

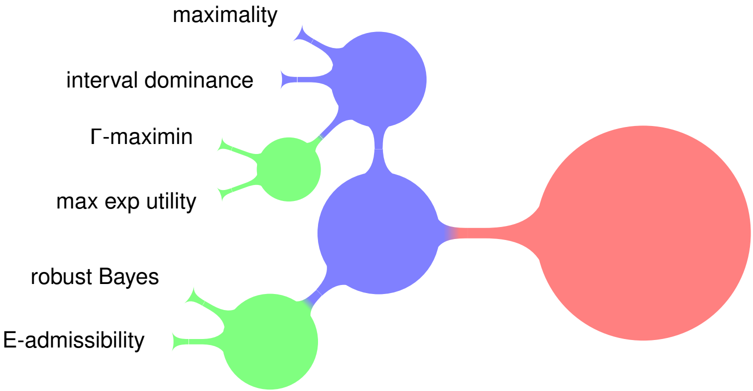
Durham University

15th September 2009

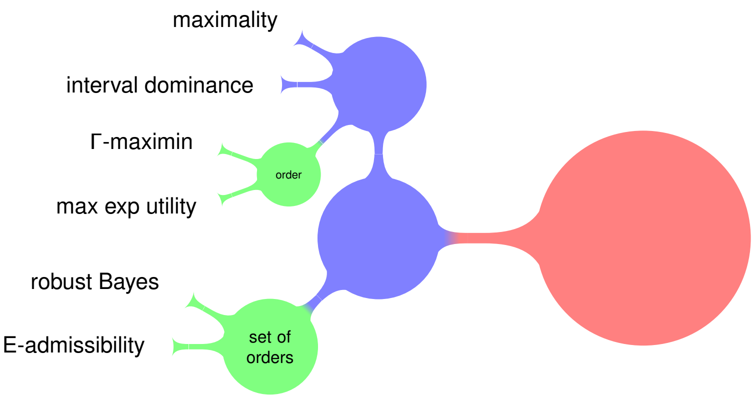
Outline

Outline

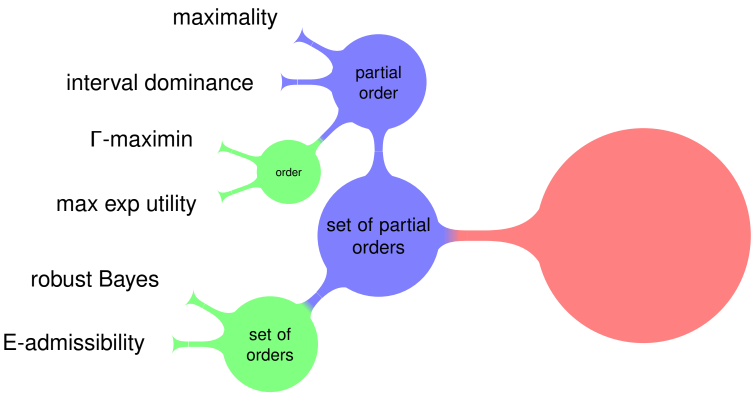
Problem Description



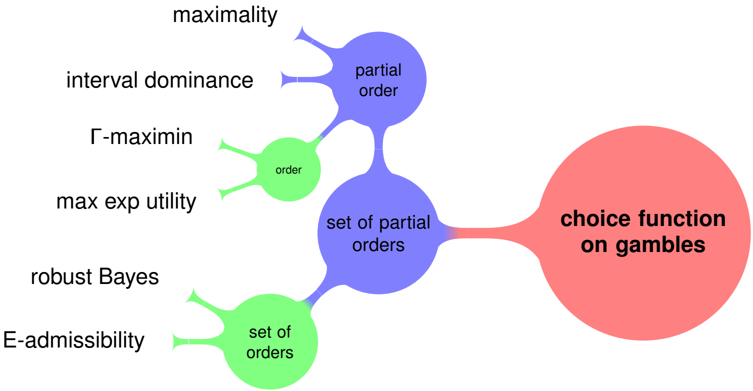
Problem Description



Problem Description



Problem Description



Outline

Gambles and Choice Functions

Definition

A gamble is an uncertain reward, i.e. a mapping from the possibility space Ω to the reward set $\mathcal{R}$.

"probabilityless (horse-)lottery"

Gambles and Choice Functions

Definition

A gamble is an uncertain reward, i.e. a mapping from the possibility space Ω to the reward set $\mathcal{R}$.

"probabilityless (horse-)lottery"

Definition

A choice function opt selects, for any set of gambles $\mathcal{X}$ and event A, a subset of $\mathcal{X}$:

$$\emptyset \neq \mathrm{opt}(\mathcal{X}|A) \subseteq \mathcal{X}$$

Gambles and Choice Functions

Definition

A gamble is an uncertain reward, i.e. a mapping from the possibility space Ω to the reward set $\mathcal{R}$.

"probabilityless (horse-)lottery"

Definition

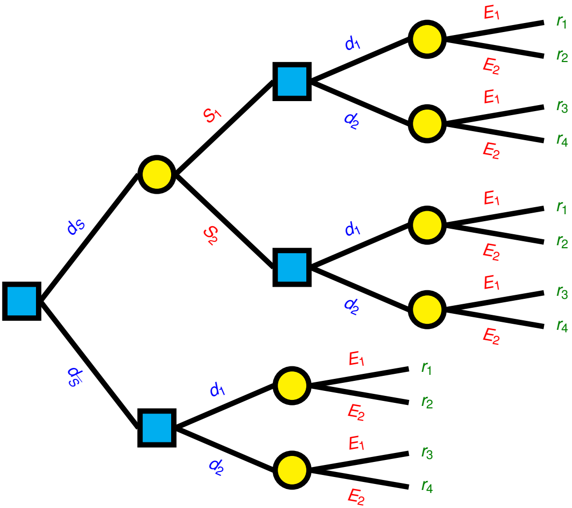
A choice function opt selects, for any set of gambles $\mathcal{X}$ and event A, a subset of $\mathcal{X}$:

$$\emptyset \neq \text{opt}(\mathcal{X}|A) \subseteq \mathcal{X}$$

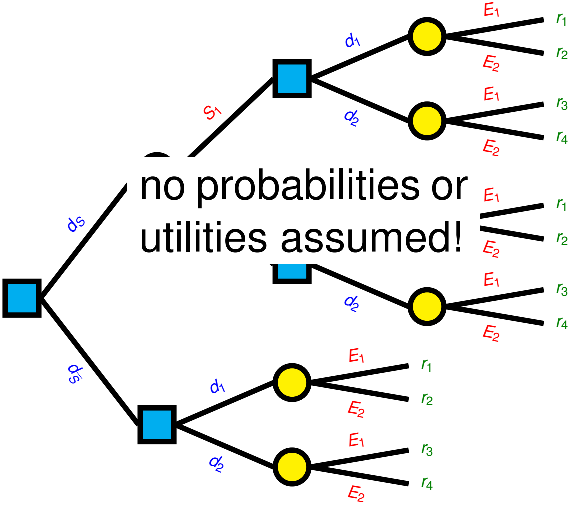
How to solve sequential decision problems
with a choice function?

Outline

Decision Trees: Example

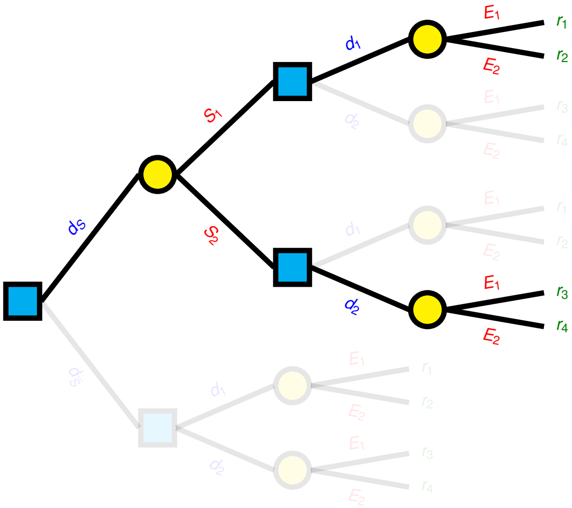


Decision Trees: Example

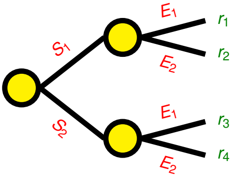


no probabilities or utilities assumed!

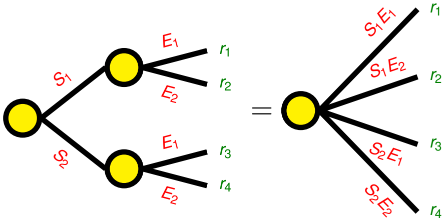
Decision Trees: Normal Form Decisions



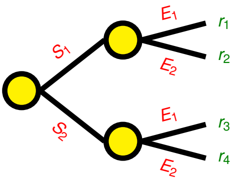
Decision Trees: Gambles



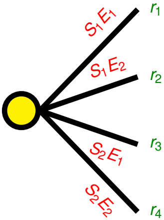
Decision Trees: Gambles



Decision Trees: Gambles

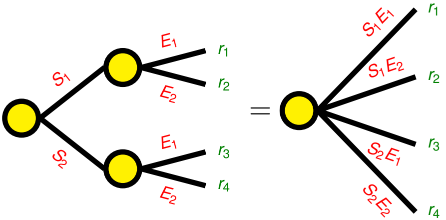


$=$



$$= X(\omega) = \begin{cases} r_1 & \text{if } \omega \in S_1E_1 \\ r_2 & \text{if } \omega \in S_1E_2 \\ r_3 & \text{if } \omega \in S_2E_1 \\ r_4 & \text{if } \omega \in S_2E_2 \end{cases}$$

Decision Trees: Gambles



$$= X(\omega) = \begin{cases} r_1 & \text{if } \omega \in S_1E_1 \\ r_2 & \text{if } \omega \in S_1E_2 \\ r_3 & \text{if } \omega \in S_2E_1 \\ r_4 & \text{if } \omega \in S_2E_2 \end{cases}$$

Observation

Every normal form decision induces a gamble.

Decision Trees: Normal Form Solution

Definition

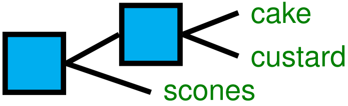
A normal form solution of a decision tree is a set of these normal form decisions.

Decision Trees: Normal Form Solution

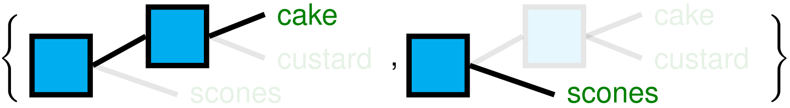
Definition

A normal form solution of a decision tree is a set of these normal form decisions.

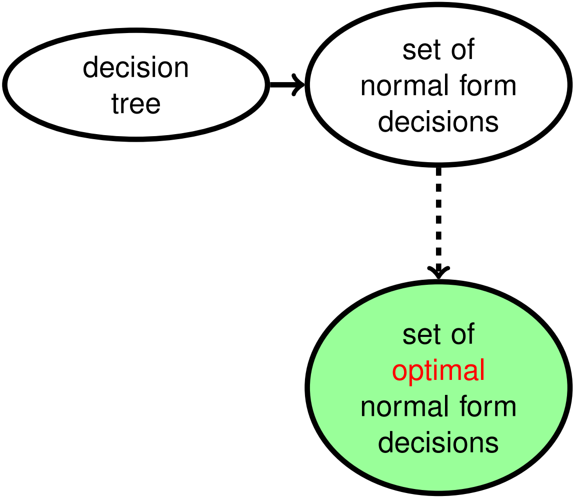
for example



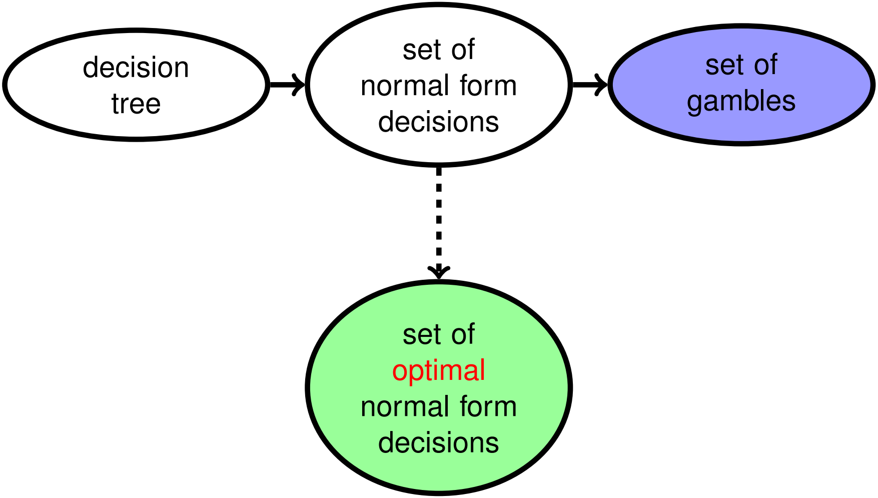
could have as normal form solution



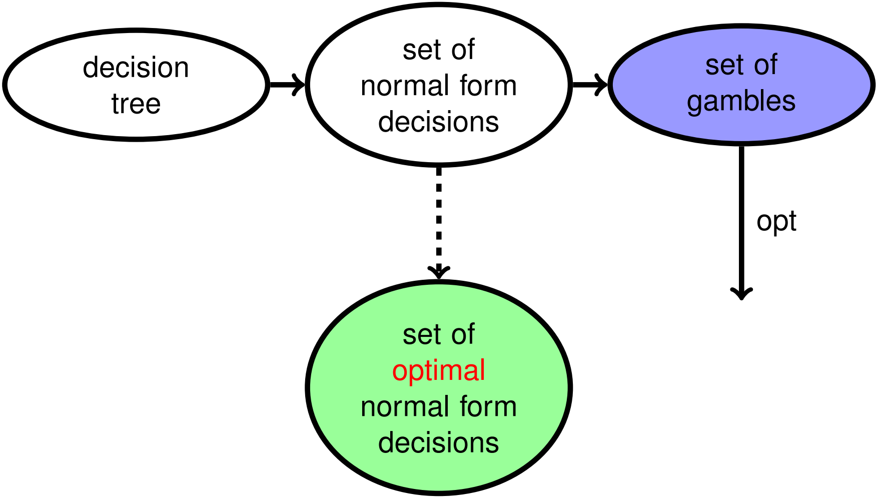
Decision Trees: Normal Form Solution Induced By opt



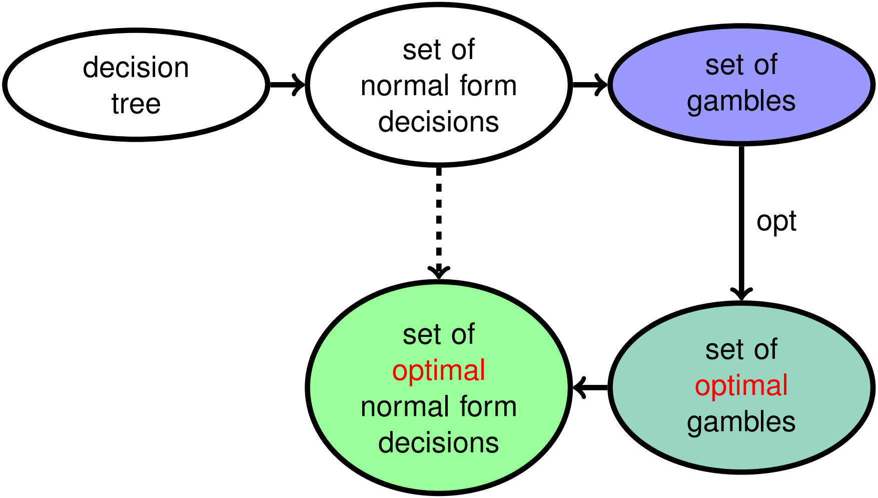
Decision Trees: Normal Form Solution Induced By opt



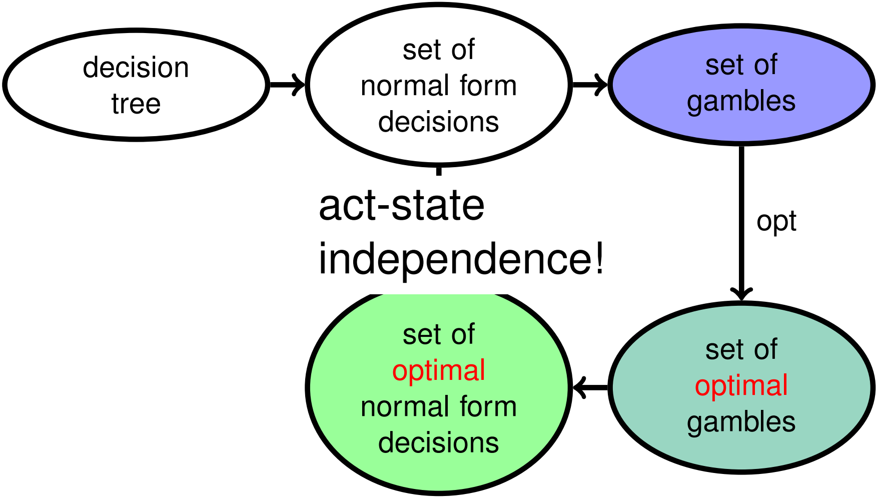
Decision Trees: Normal Form Solution Induced By opt



Decision Trees: Normal Form Solution Induced By opt



Decision Trees: Normal Form Solution Induced By opt

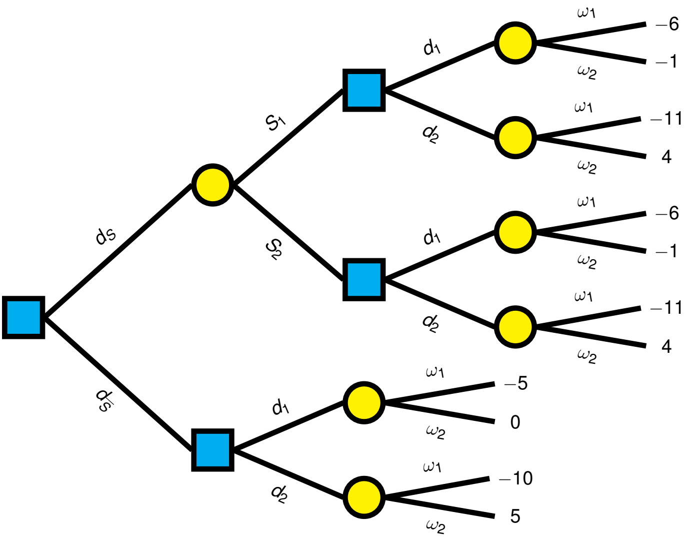


Normal Form Solution Induced By opt

Problem

- The set of normal form gambles grows very quickly with tree size
- Imprecise probability choice functions with the most attractive properties are also the most difficult to compute

Example



Outline

Backward Induction

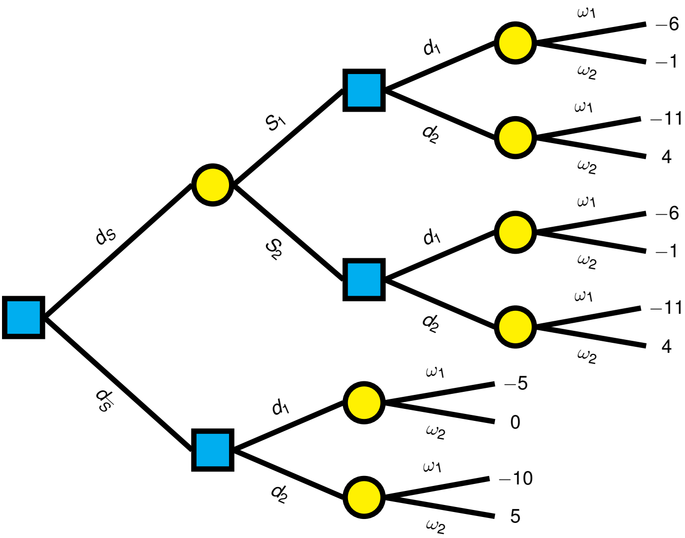
- Idea of backward induction: use the solutions of subtrees to eliminate many options in the full tree
- For factual choice functions, this works straightforwardly
- For counterfactual total preorders, there is usually no useful backward induction method
- For choice functions corresponding to partial orders (maximality, interval dominance) or more complicated choice functions (E-admissibility) there are several possibilities...

Outline

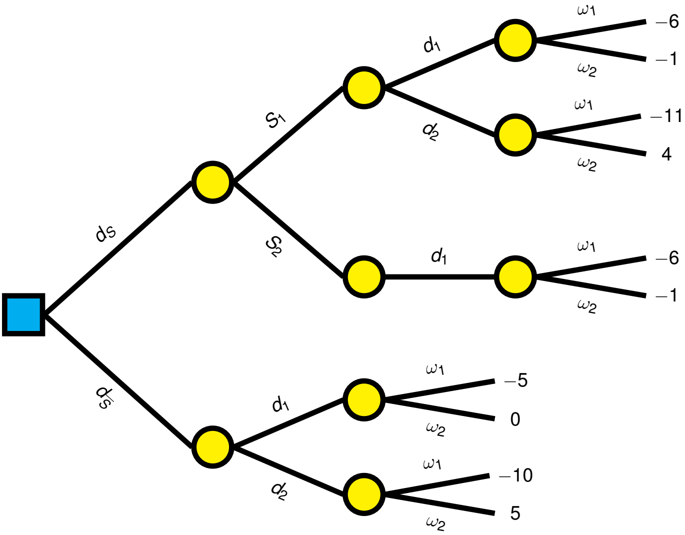
Seidenfeld 1988

- This is an extensive form solution
- Solve the subtrees at the ultimate decision nodes (only one decision, no sequential aspect)
- If a set of options remains at an ultimate decision node, we do not know what we will choose at this node. Assume maximum imprecision about this choice
- Then the choice at the next layer of decision nodes becomes non-sequential in nature

Example



Example



Problem?

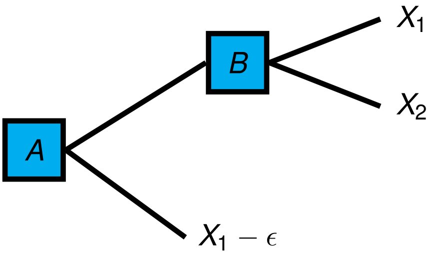


Figure: Variation of Hammond's example

Outline

Kikuti et al 2005

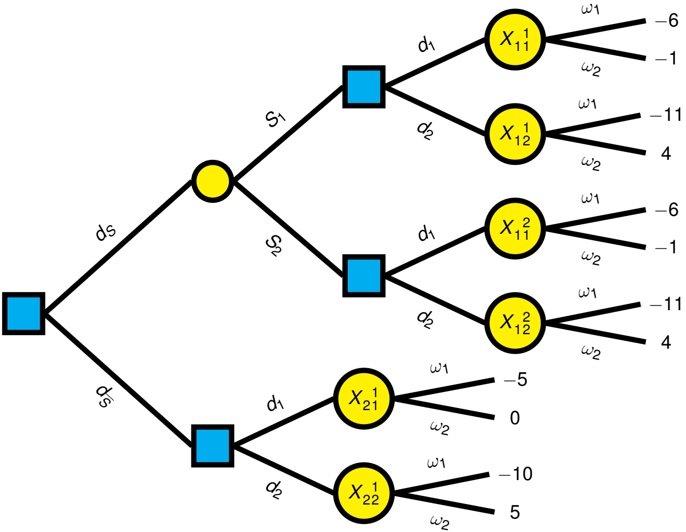
Algorithm

- Solve the subtrees at ultimate decision nodes
- Move to the previous stage of decision nodes
- Consider the normal form decisions at these nodes, but only those involving normal form decisions found optimal at the previous step
- Find the optimal subset of these normal form decisions

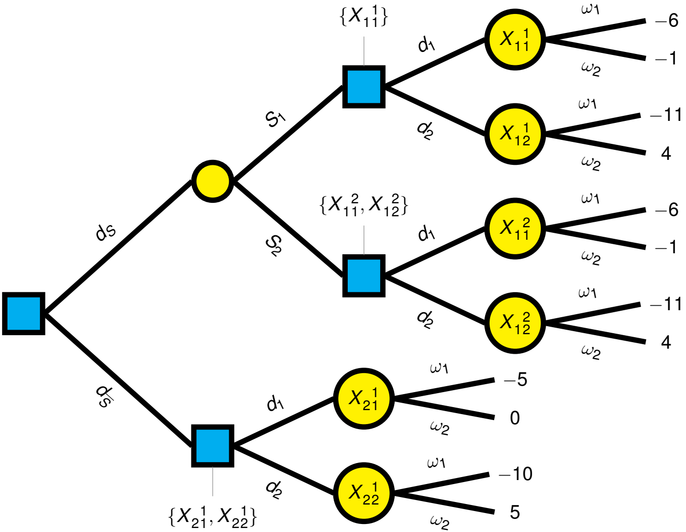
Solution

- At each decision node we end up with a set of optimal normal form decisions
- Proposed solution: at each decision node, allowed to choose any decision involved in an optimal normal form decision at this node

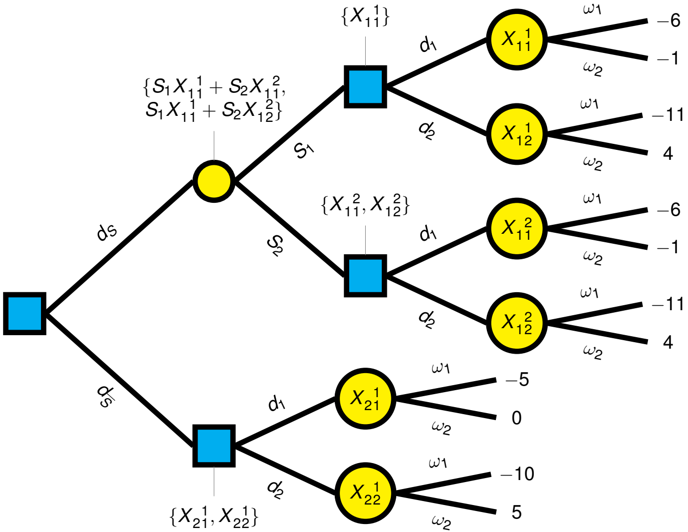
Example


d_S
$d_{\bar{S}}$
S_1
S_2
d_1
d_2
d_1
d_2
d_1
d_2
X_{11}^1
X_{12}^1
X_{11}^2
X_{12}^2
X_{21}^1
X_{22}^1
ω_1
ω_2
−6
−1
−11
4
−6
−1
−11
4
−5
0
−10
5

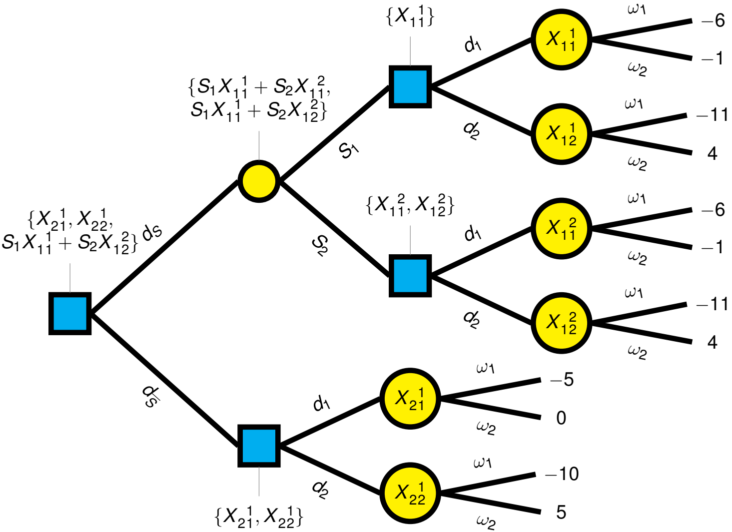
Example


$\{X_{11}^1\}$
$\{X_{11}^2, X_{12}^2\}$
$\{X_{21}^1, X_{22}^1\}$
d_S
d_S
S_1
S_2
d_1
d_2
X_{11}^1
X_{12}^1
X_{11}^2
X_{12}^2
X_{21}^1
X_{22}^1
ω_1
ω_2
-6
-1
-11
4
-6
-1
-11
4
-5
0
-10
5

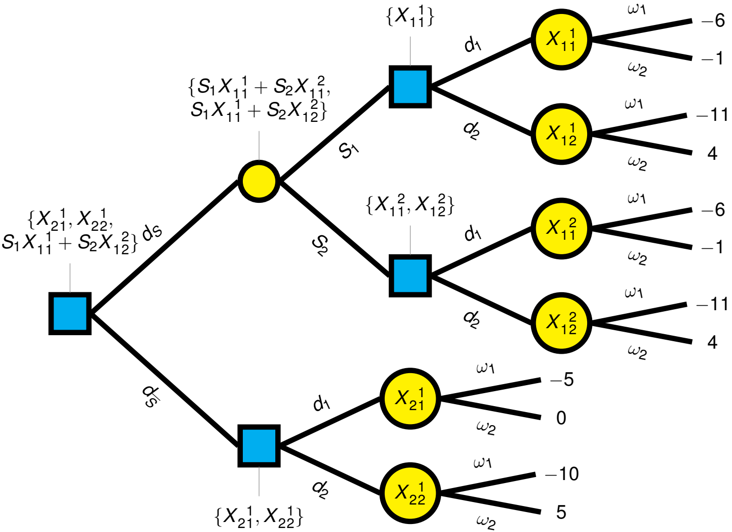
Example


$\{X_{11}^1\}$
$\{S_1X_{11}^1 + S_2X_{11}^2, S_1X_{11}^1 + S_2X_{12}^2\}$
$\{X_{11}^2, X_{12}^2\}$
$\{X_{21}^1, X_{22}^1\}$
d_S
$d_{\bar{S}}$
S_1
S_2
d_1
d_2
X_{11}^1
X_{12}^1
X_{11}^2
X_{12}^2
X_{21}^1
X_{22}^1
ω_1
ω_2
-6
-1
-11
4
-6
-1
-11
4
-5
0
-10
5

Example


$\{X_{11}^1\}$
$\{S_1X_{11}^1 + S_2X_{11}^2, S_1X_{11}^1 + S_2X_{12}^2\}$
$\{X_{11}^2, X_{12}^2\}$
$\{X_{21}^1, X_{22}^1, S_1X_{11}^1 + S_2X_{12}^2\}$
$\{X_{21}^1, X_{22}^1\}$
d_S
S_1
S_2
d_1
d_2
X_{11}^1
X_{12}^1
X_{11}^2
X_{12}^2
X_{21}^1
X_{22}^1
ω_1
ω_2
-6
-1
-11
4
-6
-1
-11
4
-5
0
-10
5

Problem?


$\{X_{11}^1\}$
$\{S_1X_{11}^1 + S_2X_{11}^2,$
$S_1X_{11}^1 + S_2X_{12}^2\}$
$\{X_{11}^2, X_{12}^2\}$
$\{X_{21}^1, X_{22}^1,$
$S_1X_{11}^1 + S_2X_{12}^2\}$
$\{X_{21}^1, X_{22}^1\}$
d_S
d_S
S_1
S_2
d_1
d_2
ω_1
ω_2
X_{11}^1
X_{12}^1
X_{11}^2
X_{12}^2
X_{21}^1
X_{22}^1
-6
-1
-11
4
-6
-1
-11
4
-5
0
-10
5

Outline

Huntley and Troffaes

- We can use the algorithm in a different way
- At the initial decision node, we have a set of “optimal” normal form decisions
- Let’s use them as our normal form solution
- This will eliminate the previous strange behaviour

Even better!

- For some choice function, this “optimal” set is nothing other than the canonical normal form solution

Necessary and Sufficient Conditions

Backward Conditioning Property

If $AX = AY$ and $\{X, Y\} \subseteq \mathcal{X}$, then $X \in \mathrm{opt}(\mathcal{X}|A) \iff Y \in \mathrm{opt}(\mathcal{X}|A)$ (subject to some technicalities)

Path Independence

$$\mathrm{opt}\left(\bigcup_{i=1}^{n} \mathcal{X}_i \middle| A\right) = \mathrm{opt}\left(\bigcup_{i=1}^{n} \mathrm{opt}(\mathcal{X}_i|A) \middle| A\right)$$

Backward Mixture Property

$$\mathrm{opt}\left(\{AX + \overline{A}Z \colon X \in \mathcal{X}\}|B\right) \subseteq A\,\mathrm{opt}(\mathcal{X}|A \cap B) \oplus \overline{A}Z$$

How useful is this?

- The choice function is applied at every stage
- If few options are deleted the process takes even longer than solving directly
- Note: Result still holds if choice function is only applied from time to time
- There are also possibilities to save time, especially if the structure of the tree is suitable

Approximation Theorem

- Suppose that:
  - $\text{opt}_1 \subseteq \text{opt}_2$
  - opt_1 satisfies the conditions
  - opt_2 does not
- We can apply the algorithm using opt_2 and then apply opt_1 at the end
- Could be useful if opt_2 is much easier to compute but still eliminates most non-optimal gambles
- Note: can apply opt_1 after opt_2 at any stage of the algorithm and nothing changes

Special Structures

- With particular structures, backward induction may work under weaker conditions, and be easier to perform.

Example

- Markov Decision Process with act-state independence
- Use maximality or E-admissibility
- $\underline{P}(\cdot) = \underline{P}(\underline{P}(\cdot|\mathcal{A}))$
- Only need solve local problems at each stage
- So: no need to ever compare too many gambles
- This generalises to a larger class of decision problems (need some symmetry, rewards at each stage depend on last choice and state history but not decision history...)
- For Γ-maximin: locality does not work but a form of backward induction does (Satia and Lave)

Relationship with Factuality

- If opt is factual then backward induction works
- If backward induction works then opt may not be factual
- In fact, backward induction implies that local solutions are supersets of the global solution
- Interpretation: knowing counterfactual information refines one's decisions

Conclusions

- Backward induction can still be used for some counterfactual choice functions
- Several possible schemes available
- So far, limited work to make these methods practical
- Question: which approach is "better"?

Nathan Huntley and Matthias Troffaes.
Normal form backward induction for decision trees under arbitrary choice functions.
Submitted.

D. Kikuti, F. Cozman, and C.P. de Campos.
Partially ordered preferences in decision trees: Computing strategies with imprecision in probabilities.
In R. Brafman and U. Junker, editors, *IJCAI-05 Multidisciplinary Workshop on Advances in Preference Handling*, pages 118–123, 2005.

T. Seidenfeld.
Decision theory without ‘independence’ or without ‘ordering’: What is the difference?
Economics and Philosophy, 4:267–290, 1988.