Machine Intelligence

Lecture 11: Planning

Thomas Dyhre Nielsen

Aalborg University

Tentative course overview

Topics:

- Introduction
- Search-based methods
- Constrained satisfaction problems
- Logic-based knowledge representation
- Representing domains endowed with uncertainty.
- Bayesian networks
- Inference in Bayesian networks
- Machine learning: classification
- Machine learning: Clustering
- **Planning**
- Multi-agent systems

Utility

A small quiz

Which of the following two lotteries would you prefer?:

- Lottery $A = [\$1\text{mill.}]$,
- Lottery $B = 0.1[\$5\text{mill.}] + 0.89[\$1\text{mill.}] + 0.01[\$0]$.

A small quiz

Which of the following two lotteries would you prefer?:

- Lottery $A = [\$1\text{mill.}]$,
- Lottery $B = 0.1[\$5\text{mill.}] + 0.89[\$1\text{mill.}] + 0.01[\$0]$.

What about these two?:

- Lottery $C = 0.11[\$1\text{mill.}] + 0.89[\$0]$,
- Lottery $D = 0.1[\$5\text{mill.}] + 0.9[\$0]$.

Preferences and Lotteries

Values with certainty

What do you prefer

- \$100 or \$1000000 ?
- A 4 or 10 grade in the exam?

Lotteries

A **lottery** is a probability distribution over **outcomes**. E.g.

$$[0.4 : \$100,\ 0.6 : -\$20]$$

means: you win \$100 with probability 0.4, and loose \$ 20 with probability 0.6.

$$[0.3 : 00,\ 0.5 : 7,\ 0.2 : 10]$$

means: with probability 0.3 you get a 00, with probability 0.5 a 7, and with probability 0.2 a 10.

Values with certainty

What do you prefer

- \$100 or \$1000000 ?
- A 4 or 10 grade in the exam?

Lotteries

A **lottery** is a probability distribution over **outcomes**. E.g.

$$[0.4 : \$100,\ 0.6 : -\$20]$$

means: you win \$100 with probability 0.4, and loose \$ 20 with probability 0.6.

$$[0.3 : 00,\ 0.5 : 7,\ 0.2 : 10]$$

means: with probability 0.3 you get a 00, with probability 0.5 a 7, and with probability 0.2 a 10.

Values with uncertainty

What do you prefer

- [1:\$1000000] or [0.5:\$ 0, 0.5:\$2100000] ?

Values with certainty

What do you prefer

- \$100 or \$1000000 ?
- A 4 or 10 grade in the exam?

Lotteries

A **lottery** is a probability distribution over **outcomes**. E.g.

$$[0.4 : \$100,\ 0.6 : -\$20]$$

means: you win \$100 with probability 0.4, and loose \$ 20 with probability 0.6.

$$[0.3 : 00,\ 0.5 : 7,\ 0.2 : 10]$$

means: with probability 0.3 you get a 00, with probability 0.5 a 7, and with probability 0.2 a 10.

Values with uncertainty

What do you prefer

- [1:\$1000000] or [0.5:\$ 0, 0.5:\$2100000] ?
- [0.4:00, 0.1:7, 0.5:10] or [0.1:00, 0.8:7, 0.1:10]?

Utilities

Typically:

$$[1{:}\$1000000] \text{ is preferred over } [0.5{:}\$\,0,\ 0.5{:}\$2100000]$$

Thus: preferences between lotteries with "money outcomes" are not always determined by **expected monetary value**.

Typically:

[1:$1000000]is preferred over [0.5:$ 0, 0.5:$2100000]

Thus: preferences between lotteries with “money outcomes” are not always determined by **expected monetary value**.

Preferences from utilities

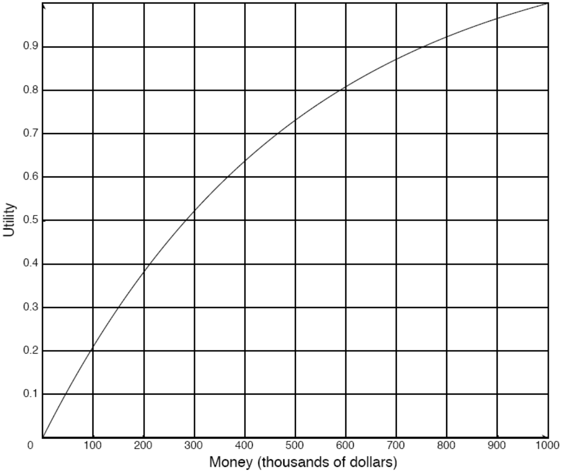
The following is a classical result:

> *If preferences between lotteries obey a certain set of plausible rules, then there exists an assignment of real numbers (**utilities**) to all outcomes, such that one lottery is preferred over another if and only if it has a higher **expected utility**.*

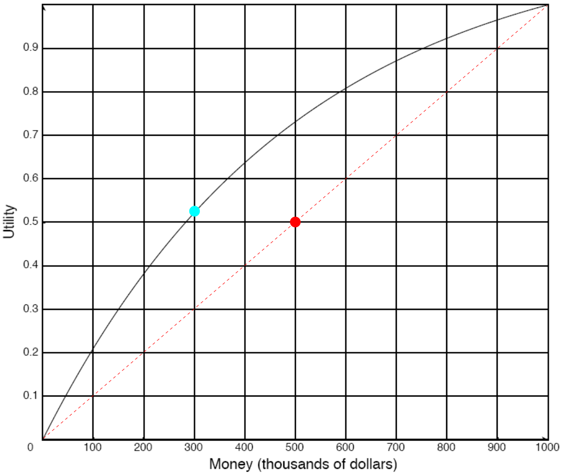
Example

				Expected Utility
Outcomes:	00	7	10	
Utilities:	-5	5	10	
Lottery 1:	0.4	0.1	0.5	3.5
Lottery 2:	0.1	0.8	0.1	4.5

Also "money outcomes" have a utility. Utility function of a **risk-averse** agent:

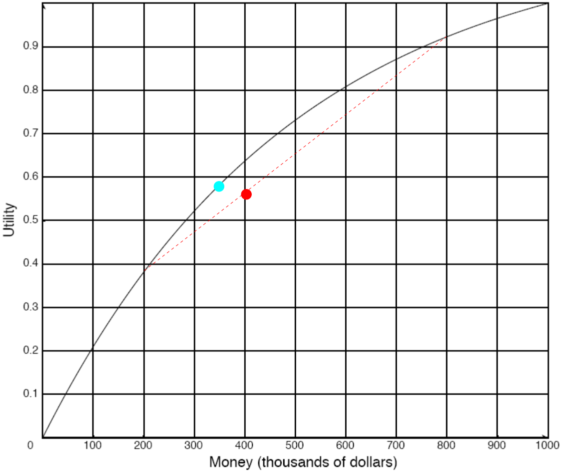


Also "money outcomes" have a utility. Utility function of a **risk-averse** agent:



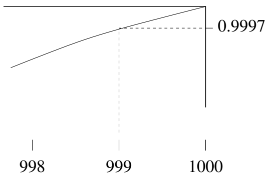
- Sure \$ 300k preferred over [0.5:\$ 0, 0.5:\$ 1000k]

Also “money outcomes” have a utility. Utility function of a **risk-averse** agent:



- Sure $ 300k preferred over [0.5:$ 0, 0.5:$ 1000k]
- Sure $ 350k preferred over [2/3:$ 200,1/3:$ 800k]

Assume Utility($ 999k)=0.9997:



Then agent is indifferent between lotteries

[1:$ 999k] and [0.0003:0, 0.9997:$ 1000k]

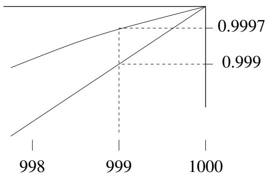
and prefers

[1:$ 999.4k] over [0.0003:0, 0.9997:$ 1000k]

Interpretation:

- right lottery: 0.03 risk of loosing a $ 1000k property
- left lottery: buying insurance against that risk for $ 600.

Assume Utility($ 999k)=0.9997:



Then agent is indifferent between lotteries

[1:$ 999k] and [0.0003:0, 0.9997:$ 1000k]

and prefers

[1:$ 999.4k] over [0.0003:0, 0.9997:$ 1000k]

Interpretation:

- right lottery: 0.03 risk of loosing a $ 1000k property
- left lottery: buying insurance against that risk for $ 600.

The *insurance company* prefers

[0.0003:0, 0.9997:$ 1000k] over [1:$ 999.4k]

$\rightsquigarrow$ the insurance company has a different utility function (near linear).

Do we maximize expected utility?

Recall:

- Lottery $A = [\$1\text{mill.}]$,
- Lottery $B = 0.1[\$5\text{mill.}] + 0.89[\$1\text{mill.}] + 0.01[\$0]$.
- Lottery $C = 0.11[\$1\text{mill.}] + 0.89[\$0]$,
- Lottery $D = 0.1[\$5\text{mill.}] + 0.9[\$0]$.

Recall:

- Lottery $A = [\$1\text{mill.}]$,
- Lottery $B = 0.1[\$5\text{mill.}] + 0.89[\$1\text{mill.}] + 0.01[\$0]$.
- Lottery $C = 0.11[\$1\text{mill.}] + 0.89[\$0]$,
- Lottery $D = 0.1[\$5\text{mill.}] + 0.9[\$0]$.

A preferred over $B \Rightarrow$

$$
\begin{aligned}
& U(\$1m) > 0.1U(\$5m) + 0.89U(\$1m) + 0.01U(\$0) \\
\Rightarrow \quad & 0.11U(\$1m) > 0.1U(\$5m) + 0.01U(\$0)
\end{aligned}
$$

Recall:

- Lottery $A = [\$1\text{mill.}]$,
- Lottery $B = 0.1[\$5\text{mill.}] + 0.89[\$1\text{mill.}] + 0.01[\$0]$.
- Lottery $C = 0.11[\$1\text{mill.}] + 0.89[\$0]$,
- Lottery $D = 0.1[\$5\text{mill.}] + 0.9[\$0]$.

A preferred over $B \Rightarrow$

$$\begin{aligned} & U(\$1m) > 0.1U(\$5m) + 0.89U(\$1m) + 0.01U(\$0) \\ \Rightarrow \quad & 0.11U(\$1m) > 0.1U(\$5m) + 0.01U(\$0) \end{aligned}$$

D preferred over $C \Rightarrow$

$$\begin{aligned} & 0.1U(\$5m) + 0.9U(\$0) > 0.11U(\$1m) + 0.89U(\$0) \\ \Rightarrow \quad & 0.1U(\$5m) + 0.01U(\$0) > 0.11U(\$1m) \end{aligned}$$

Do we maximize expected utility?

Recall:

- Lottery $A = [\$1\text{mill.}]$,
- Lottery $B = 0.1[\$5\text{mill.}] + 0.89[\$1\text{mill.}] + 0.01[\$0]$.
- Lottery $C = 0.11[\$1\text{mill.}] + 0.89[\$0]$,
- Lottery $D = 0.1[\$5\text{mill.}] + 0.9[\$0]$.

A preferred over $B \Rightarrow$

$$\begin{array}{rl} & U(\$1m) > 0.1U(\$5m) + 0.89U(\$1m) + 0.01U(\$0) \\ \Rightarrow & 0.11U(\$1m) > 0.1U(\$5m) + 0.01U(\$0) \end{array}$$

D preferred over $C \Rightarrow$

$$\begin{array}{rl} & 0.1U(\$5m) + 0.9U(\$0) > 0.11U(\$1m) + 0.89U(\$0) \\ \Rightarrow & 0.1U(\$5m) + 0.01U(\$0) > 0.11U(\$1m) \end{array}$$

Contradiction! Explanations:

- People do not maximize expected utility
- The utility of \$ 0 also depends on in which lottery \$ 0 were “won”

Factored Utility

So far: outcomes seen as unstructured states. When states are described by features, then overall utility often a combination of utility factors derived from different features.

Example

Two component utility function:

RHC	*SWC*	Utility
rhc	*swc*	5
rhc	$\overline{\textit{swc}}$	3
$\overline{\textit{rhc}}$	*swc*	0
$\overline{\textit{rhc}}$	$\overline{\textit{swc}}$	5

RLoc	*MW*	Utility
cs	*mw*	0
cs	$\overline{\textit{mw}}$	2
off	*mw*	0
off	$\overline{\textit{mw}}$	2
lab	*mw*	0
lab	$\overline{\textit{mw}}$	2
mr	*mw*	5
mr	$\overline{\textit{mw}}$	2

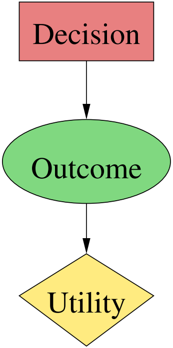
Utility of full outcome (state) is sum of utility factors:

$$U(\langle \textit{off}, \textit{rhc}, \overline{\textit{swc}}, \overline{\textit{mw}}, \textit{rhm}\rangle) = 3 + 2 = 5$$

Assumption: the utility contribution from one factor is independent of the values of other factors. E.g.: (*rhc*,*swc*) should perhaps be worth less than 5 when at the same time (*mr*,*mw*), because mail needs to be delivered first (the two utility factors are **substitutes**).

Single-Stage Decision Networks

Simple decisions can be seen as choices over lotteries. Graphical representation of study example:



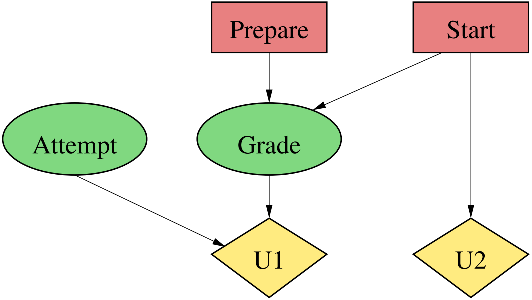
prepare_some,*prepare_all*

Decision	*Outcome* 00	7	10
prepare_some	0.4	0.1	0.5
prepare_all	0.1	0.8	0.1

Outcome	Utility
00	-5
7	5
10	10

Decisions, outcomes and utilities can all be composed of features or factors:

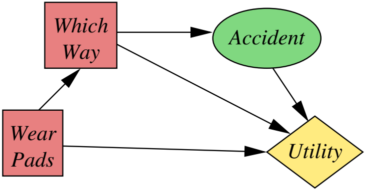
Two components of decision: prepare *some*/*all*, start preparations *sooner*/*later*
Two utility factors: utility of grade, and utility (cost) of preparation time
Outcome composed of *Grade*, *Attempt*



Graph represents:

- One utility factor depends on *Attempt* and *Grade*, another only on *Start*
- Both the *Prepare* and *Start* decision influence the probabilities for *Grade*.

Robot Example

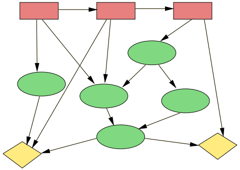


	Accident	
WhichWay	true	false
short	0.2	0.8
long	0.01	0.99

WearPads	*WhichWay*	*Accident*	Utility
true	short	true	35
true	short	false	95
true	long	true	30
true	long	false	75
false	short	true	3
false	short	false	100
false	long	true	0
false	long	false	80

Structure

A **Single-Stage Decision Network** is a directed acyclic graph with three types of nodes:



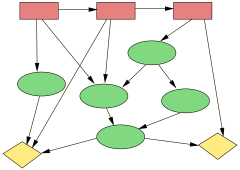
- Decision Nodes $\mathbf{D}$
- Chance Nodes $\mathbf{C}$
- Utility Nodes $\mathbf{U}$

The graph must have the following structure:

- All decision nodes are connected in one linear sequence (representing the order in which the different sub-decisions are taken)
- The only parent of a decision node is its predecessor in the order
- Chance Nodes can have Decision Node and Chance Node parents
- Utility Nodes can have Decision Node and Chance Node parents

Structure

A **Single-Stage Decision Network** is a directed acyclic graph with three types of nodes:



- Decision Nodes $\mathbf{D}$
- Chance Nodes $\mathbf{C}$
- Utility Nodes $\mathbf{U}$

The graph must have the following structure:

- All decision nodes are connected in one linear sequence (representing the order in which the different sub-decisions are taken)
- The only parent of a decision node is its predecessor in the order
- Chance Nodes can have Decision Node and Chance Node parents
- Utility Nodes can have Decision Node and Chance Node parents

Tables

- No table is associated with decision nodes (only the list of available decisions)
- A chance node is labeled with a conditional probability table that specifies for each value assignment to its parents (decision and chance nodes) a probability distribution over the domain of the chance node.
- A utility node is labeled with a utility table that specifies for each value assignment to its parents (decision and chance nodes) a utility value.

SSDN Semantics

A **possible world** ω is an assignment of values to all decision and chance variables.
An SSDN defines:

- For each assignment $\mathbf{D} = \mathbf{d}$ of values to the decision nodes a probability distribution

$$P(\omega \mid \mathbf{D} = \mathbf{d})$$

  over possible worlds.
- For each possible world ω a utility value

$$U(\omega)$$

SSDN Semantics

A **possible world** ω is an assignment of values to all decision and chance variables.
An SSDN defines:

- For each assignment $\mathbf{D} = \mathbf{d}$ of values to the decision nodes a probability distribution

  $$P(\omega \mid \mathbf{D} = \mathbf{d})$$

  over possible worlds.
- For each possible world ω a utility value

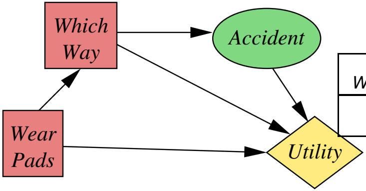
  $$U(\omega)$$

Solving an SSDN

To **solve** a single-stage decision network (or problem) means to find the decisions $\mathbf{d}$ that maximize the expected utility

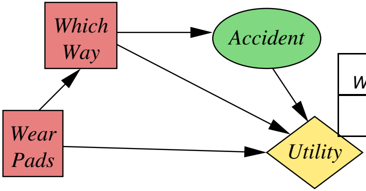
$$\mathcal{E}(U \mid \mathbf{D} = \mathbf{d}) = \sum_{\omega} U(\omega) P(\omega \mid \mathbf{D} = \mathbf{d})$$

Robot Example



WhichWay	Accident: true	Accident: false
short	0.2	0.8
long	0.01	0.99

WearPads	*WhichWay*	*Accident*	Utility
true	short	true	35
true	short	false	95
true	long	true	30
true	long	false	75
false	short	true	3
false	short	false	100
false	long	true	0
false	long	false	80



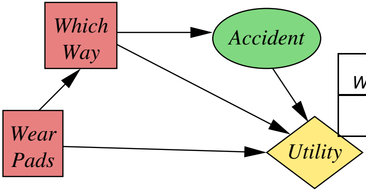
WhichWay	*Accident* true	false
short	0.2	0.8
long	0.01	0.99

WearPads	*WhichWay*	*Accident*	Utility
true	short	true	35
true	short	false	95
true	long	true	30
true	long	false	75
false	short	true	3
false	short	false	100
false	long	true	0
false	long	false	80

$P(\textit{Accident} \mid \textit{WhichWay}) \cdot U(\textit{WearPads}, \textit{WhichWay}, \textit{Accident}) =$

WearPads	*WhichWay*	*Accident*	$P(\omega \mid \mathbf{D} = \mathbf{d}) \cdot U(\omega)$
true	short	true	0.2· 35
true	short	false	0.8· 95
true	long	true	0.01· 30
true	long	false	0.99· 75
false	short	true	0.2· 3
false	short	false	0.8· 100
false	long	true	0.01· 0
false	long	false	0.99· 80

Robot Example



WhichWay	Accident true	Accident false
short	0.2	0.8
long	0.01	0.99

WearPads	*WhichWay*	*Accident*	Utility
true	short	true	35
true	short	false	95
true	long	true	30
true	long	false	75
false	short	true	3
false	short	false	100
false	long	true	0
false	long	false	80

$P(\textit{Accident} \mid \textit{WhichWay}) \cdot U(\textit{WearPads}, \textit{WhichWay}, \textit{Accident}) =$

WearPads	*WhichWay*	*Accident*	$P(\omega \mid \mathbf{D} = \mathbf{d}) \cdot U(\omega)$
true	short	true	0.2· 35
true	short	false	0.8· 95
true	long	true	0.01· 30
true	long	false	0.99· 75
false	short	true	0.2· 3
false	short	false	0.8· 100
false	long	true	0.01· 0
false	long	false	0.99· 80

$\sum_{\textit{Accident}} P(\textit{Accident} \mid \textit{WhichWay}) \cdot U(\textit{WearPads}, \textit{WhichWay}, \textit{Accident}) =$

WearPads	*WhichWay*	$\mathcal{E}(U \mid \mathbf{D} = \mathbf{d})$
true	short	0.2· 35 + 0.8· 95 = 83
true	long	0.01· 30 + 0.99· 75 = 74.55
false	short	0.2· 3 + 0.8· 100= 80.6
false	long	0.01· 0 + 0.99· 80= 79.2

Solving SSDNs in general

To **solve** a single-stage decision network (or problem) means to find the decisions $\mathbf{d}$ that maximize the expected utility

$$\mathcal{E}(U \mid \mathbf{D} = \mathbf{d}) = \sum_{\omega} U(\omega) P(\omega \mid \mathbf{D} = \mathbf{d})$$

To **solve** a single-stage decision network (or problem) means to find the decisions $\mathbf{d}$ that maximize the expected utility

$$\mathcal{E}(U \mid \mathbf{D} = \mathbf{d}) = \sum_{\omega} U(\omega) P(\omega \mid \mathbf{D} = \mathbf{d})$$

Solving by (Chance) variable elimination

Let $P(C_1 \mid \textit{par}(C_1)), \ldots, P(C_n \mid \textit{par}(C_n))$ be the conditional probability tables associated with the chance nodes, and $U_1(\textit{par}(U_1)), \ldots, U_k(\textit{par}(U_k))$ the utility tables of the utility nodes. Then

- For each $j = 1, \ldots, k$:
  $$\prod_{i=1}^{n} P(C_i \mid \textit{par}(C_i)) U_j(\textit{par}(U_j))$$
  is a table in the variables $\mathbf{D}, \mathbf{C}$. Each row in the table corresponds to a possible world $\omega \sim (\mathbf{D} = \mathbf{d}, \mathbf{C} = \mathbf{c})$. The entry for that row is equal to $U_j(\omega) P(\omega \mid \mathbf{D} = \mathbf{d})$.

Solving SSDNs in general

To **solve** a single-stage decision network (or problem) means to find the decisions $\mathbf{d}$ that maximize the expected utility

$$\mathcal{E}(U \mid \mathbf{D} = \mathbf{d}) = \sum_{\omega} U(\omega) P(\omega \mid \mathbf{D} = \mathbf{d})$$

Solving by (Chance) variable elimination

Let $P(C_1 \mid \mathit{par}(C_1)), \ldots, P(C_n \mid \mathit{par}(C_n))$ be the conditional probability tables associated with the chance nodes, and $U_1(\mathit{par}(U_1)), \ldots, U_k(\mathit{par}(U_k))$ the utility tables of the utility nodes. Then

- For each $j = 1, \ldots, k$:

  $$\prod_{i=1}^{n} P(C_i \mid \mathit{par}(C_i)) U_j(\mathit{par}(U_j))$$

  is a table in the variables $\mathbf{D}, \mathbf{C}$. Each row in the table corresponds to a possible world $\omega \sim (\mathbf{D} = \mathbf{d}, \mathbf{C} = \mathbf{c})$. The entry for that row is equal to $U_j(\omega) P(\omega \mid \mathbf{D} = \mathbf{d})$.
- The expected utility factor U_j of decision $\mathbf{D} = \mathbf{d}$ is obtained by summing the chance variables out of that table:

  $$\mathcal{E}(U_j \mid \mathbf{D} = \mathbf{d}) = \sum_{\mathbf{C}} \prod_{i=1}^{n} P(C_i \mid \mathit{par}(C_i)) U_j(\mathit{par}(U_j))$$

Solving SSDNs in general

To **solve** a single-stage decision network (or problem) means to find the decisions $\mathbf{d}$ that maximize the expected utility

$$\mathcal{E}(U \mid \mathbf{D} = \mathbf{d}) = \sum_{\omega} U(\omega) P(\omega \mid \mathbf{D} = \mathbf{d})$$

Solving by (Chance) variable elimination

Let $P(C_1 \mid \mathit{par}(C_1)), \ldots, P(C_n \mid \mathit{par}(C_n))$ be the conditional probability tables associated with the chance nodes, and $U_1(\mathit{par}(U_1)), \ldots, U_k(\mathit{par}(U_k))$ the utility tables of the utility nodes. Then

- For each $j = 1, \ldots, k$:

  $$\prod_{i=1}^{n} P(C_i \mid \mathit{par}(C_i)) U_j(\mathit{par}(U_j))$$

  is a table in the variables $\mathbf{D}, \mathbf{C}$. Each row in the table corresponds to a possible world $\omega \sim (\mathbf{D} = \mathbf{d}, \mathbf{C} = \mathbf{c})$. The entry for that row is equal to $U_j(\omega) P(\omega \mid \mathbf{D} = \mathbf{d})$.
- The expected utility factor U_j of decision $\mathbf{D} = \mathbf{d}$ is obtained by summing the chance variables out of that table:

  $$\mathcal{E}(U_j \mid \mathbf{D} = \mathbf{d}) = \sum_{\mathbf{C}} \prod_{i=1}^{n} P(C_i \mid \mathit{par}(C_i)) U_j(\mathit{par}(U_j))$$

- The decisions that maximize the expected utility can be found from the sum of the expected utility factors:

  $$\mathit{maxEU}(\mathbf{D}) = \max_{\mathbf{D}=\mathbf{d}} \sum_{j=1}^{k} \sum_{\mathbf{C}} \prod_{i=1}^{n} P(C_i \mid \mathit{par}(C_i)) U_j(\mathit{par}(U_j))$$

Sequential Decisions

Generalization

SSDNs are generalized to sequential decision problems by:

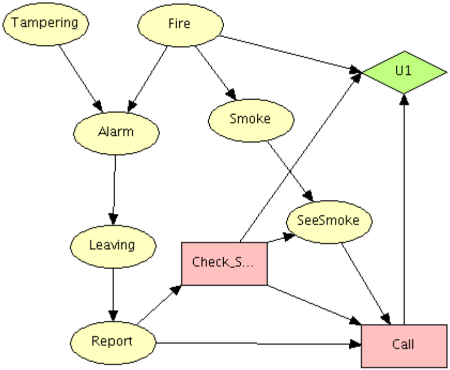
- several decisions are taken (in a fixed order)
- some chance variables may be *observed* before the next decision is taken

Examples

- Doctor first decides which test to perform, then observes test outcome, then decides which treatment to prescribe
- Before we decide to take the umbrella with us, we observe the weather forecast
- A company first decides whether to develop a certain product, then observes the customer reaction in a test market, then decides whether to go into full production.

Fire alarm example extended with

- two decisions: *CheckSmoke* $\in \{yes, no\}$, *Call* $\in \{call, do\_not\_call\}$
- one more random variable *SeeSmoke* $\in \{yes, no\}$
- a utility function depending on *Fire*,*CheckSmoke*,*Call* (composed of two factors: one depending on *CheckSmoke*, one on *Fire* and *Call*)



CheckSmoke	*Fire*	*Call*	Utility
yes	yes	call	-220
yes	yes	do_not_call	-5020
yes	no	call	-220
yes	no	do_not_call	-20
no	yes	call	-200
no	yes	do_not_call	-5000
no	no	call	-200
no	no	do_not_call	0

We assume that the decision maker doesn't forget: the *no-forgetting assumption*

Chance variables:

- $OH_0, OH_1, OH_2, MH_0, MH_1, MH_2 \in \{$*nothing*, *ace*, *2 of a kind*, *2 aces*, *flush*, *straight*, *straight flush*$\}$ represent my and opponent's hand after 0,1,2 card exchanges
- $BH \in \{$*me*, *opponent*, *draw*$\}$ represents who has the better hand after the exchanges
- $OFC \in \{0, 1, 2, 3\}$, $OSC \in \{0, 1, 2\}$ represent how many cards opponent changes in first/second exchange

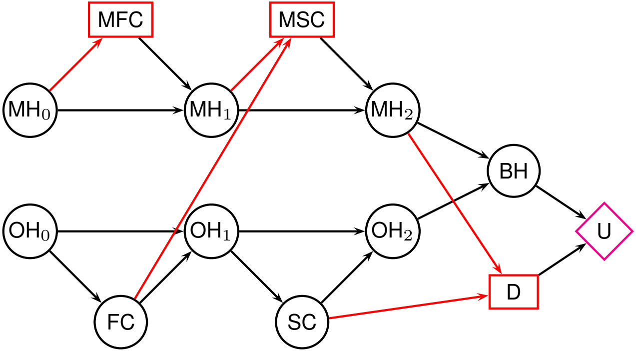
Decision variables:

- $MFC \in \{0, 1, 2, 3\}$, $MSC \in \{0, 1, 2\}$ represent how many cards I exchange in first/second exchange
- $D \in \{$*fold*, *call*$\}$ represents whether I decide to fold or call after second exchange

Utility function:

	D	
BH	*fold*	*call*
me	-1	2
opponent	-1	-2
draw	-1	0

Possible decision model



Induced order on observations and decisions:

$$\{MH_0\} \prec MFC \prec \{MH_1, OFC\} \prec MSC \prec \{MH_2, OSC\} \prec D \prec \{OH_0, OH_1, OH_2, BH\}$$

Decision Function

A **Decision Function** for a decision node D is an assignment of a decision d to each possible configuration of D's parents.

Example:

Report	*CheckSmoke*	*SeeSmoke*	*Call*
yes	yes	yes	call
yes	yes	no	do_not_call
yes	no	yes	do_not_call
yes	no	no	do_not_call
no	yes	yes	call
no	yes	no	do_not_call
no	no	yes	do_not_call
no	no	no	do_not_call

Policies

A **Policy** π consists of one decision function for each decision node.

- General strategy for actions (decisions), taking into account the possible (uncertain) effects of previous actions

Expected Utility

As before: possible worlds ω are assignments for all decision and chance variables.

- A policy π defines a probability distribution

$$P(\omega \mid \pi)$$

  over possible worlds:
  - if ω contains assignments to a decision node D and its parents which is not consistent with the decision function for D: $P(\omega \mid \pi) = 0$.
  - Otherwise: $P(\omega \mid \pi)$ is the product of all conditional probability values for the assignments to chance nodes C, given the assignment to the parents of C
- Each possible world has a utility

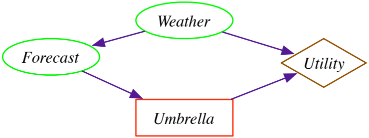
$$U(\omega)$$

- Obtain expected utility of a policy

$$\mathcal{E}(U \mid \pi) = \sum_{\omega} U(\omega) P(\omega \mid \pi)$$

Optimal Policy

An **optimal policy** is a policy with maximal expected utility (among all possible policies).

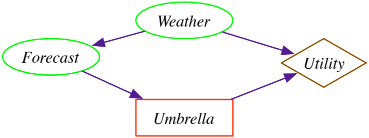


Initial distributions:

Weather	Value
norain	0.7
rain	0.3

Weather	*Umb*	Value
norain	take	20
norain	leave	100
rain	take	70
rain	leave	0

Weather	*Fcast*	Value
norain	sunny	0.7
norain	cloudy	0.2
norain	rainy	0.1
rain	sunny	0.15
rain	cloudy	0.25
rain	rainy	0.6



Initial distributions:

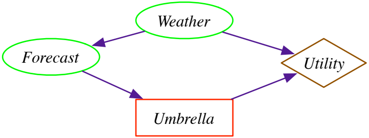
Weather	Value
norain	0.7
rain	0.3

Weather	Umb	Value
norain	take	20
norain	leave	100
rain	take	70
rain	leave	0

Weather	Fcast	Value
norain	sunny	0.7
norain	cloudy	0.2
norain	rainy	0.1
rain	sunny	0.15
rain	cloudy	0.25
rain	rainy	0.6

Combine the factors

Umb	Weather	Fcast	Value
take	norain	sunny	9.8
take	norain	cloudy	2.8
take	norain	rainy	1.4
take	rain	sunny	3.15
take	rain	cloudy	5.25
take	rain	rainy	12.6
leave	norain	sunny	49
leave	norain	cloudy	14
leave	norain	rainy	7
leave	rain	sunny	0
leave	rain	cloudy	0
leave	rain	rainy	0



Initial distributions:

Weather	Value
norain	0.7
rain	0.3

Weather	*Umb*	Value
norain	take	20
norain	leave	100
rain	take	70
rain	leave	0

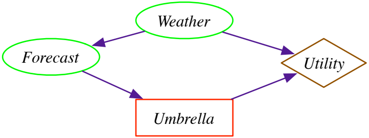
Weather	*Fcast*	Value
norain	sunny	0.7
norain	cloudy	0.2
norain	rainy	0.1
rain	sunny	0.15
rain	cloudy	0.25
rain	rainy	0.6

Combine the factors

Umb	*Weather*	*Fcast*	Value
take	norain	sunny	9.8
take	norain	cloudy	2.8
take	norain	rainy	1.4
take	rain	sunny	3.15
take	rain	cloudy	5.25
take	rain	rainy	12.6
leave	norain	sunny	49
leave	norain	cloudy	14
leave	norain	rainy	7
leave	rain	sunny	0
leave	rain	cloudy	0
leave	rain	rainy	0

$\Rightarrow \sum_{Weather}$

Umb	*Fcast*	Value
take	sunny	12.95
take	cloudy	8.05
take	rainy	14
leave	sunny	49
leave	cloudy	14
leave	rainy	7



Initial distributions:

Weather	Value
norain	0.7
rain	0.3

Weather	*Fcast*	Value
norain	sunny	0.7
norain	cloudy	0.2
norain	rainy	0.1
rain	sunny	0.15
rain	cloudy	0.25
rain	rainy	0.6

Weather	*Umb*	Value
norain	take	20
norain	leave	100
rain	take	70
rain	leave	0

Marginalising out ***Umb***

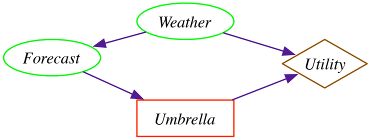
Umb	*Fcast*	Value
take	sunny	12.95
take	cloudy	8.05
take	rainy	14
leave	sunny	49
leave	cloudy	14
leave	rainy	7

$\max_{Umb} f$:

Fcast	Val
sunny	49.0
cloudy	14.0
rainy	14.0

$\arg\max_{Umb} f$:

Fcast	*Umb*
sunny	leave
cloudy	leave
rainy	take



Initial distributions:

Weather	Value
norain	0.7
rain	0.3

Weather	*Umb*	Value
norain	take	20
norain	leave	100
rain	take	70
rain	leave	0

Weather	*Fcast*	Value
norain	sunny	0.7
norain	cloudy	0.2
norain	rainy	0.1
rain	sunny	0.15
rain	cloudy	0.25
rain	rainy	0.6

Marginalising out *Umb*

Umb	*Fcast*	Value
take	sunny	12.95
take	cloudy	8.05
take	rainy	14
leave	sunny	49
leave	cloudy	14
leave	rainy	7

$\max_{Umb} f$:

Fcast	Val
sunny	49.0
cloudy	14.0
rainy	14.0

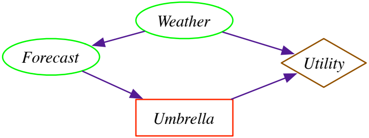
$\arg\max_{Umb} f$:

Fcast	*Umb*
sunny	leave
cloudy	leave
rainy	take

$\equiv$

Umb	*Fcast*	Value
take	sunny	0
take	cloudy	0
take	rainy	1
leave	sunny	1
leave	cloudy	1
leave	rainy	0

Solving Sequential Decision Problems I



Initial distributions:

Weather	Value
norain	0.7
rain	0.3

Weather	*Umb*	Value
norain	take	20
norain	leave	100
rain	take	70
rain	leave	0

Weather	*Fcast*	Value
norain	sunny	0.7
norain	cloudy	0.2
norain	rainy	0.1
rain	sunny	0.15
rain	cloudy	0.25
rain	rainy	0.6

Marginalising out *Umb*

Umb	*Fcast*	Value
take	sunny	12.95
take	cloudy	8.05
take	rainy	14
leave	sunny	49
leave	cloudy	14
leave	rainy	7

$\max_{Umb} f$:

Fcast	Val
sunny	49.0
cloudy	14.0
rainy	14.0

$\arg\max_{Umb} f$:

Fcast	*Umb*
sunny	leave
cloudy	leave
rainy	take

$\equiv$

Umb	*Fcast*	Value
take	sunny	0
take	cloudy	0
take	rainy	1
leave	sunny	1
leave	cloudy	1
leave	rainy	0

We now have a new decision problem with one decision less. This decision problem can be solved using the same procedure until no decisions are left!

Intuition

- Given values assigned to its parents, the last decision node can be seen as a single-stage decision.
- When all decisions following a given decision D are taken according to fixed decision rules, then D also behaves like a single-stage decision.
- Backward strategy:
  - find the decision rule for the last decision D that is not yet eliminated.
  - eliminate D by replacing it with the resulting utility factor
- Formal way of “What would I do if ...” reasoning

Variable Elimination

1. *DFs* $= \emptyset$ // Set of decision functions
2. *Fs*= all conditional probability and utility tables
3. **while** there are decision nodes
4. sum out all random variables that are not parents of a decision node
   // *Fs now contains a factor F that depends on one decision node D*
   // *and (a subset of) its parents**/
5. Add $max_D F$ to *Fs*
6. Add $arg\,max_D F$ to *DFs*
7. Sum out remaining random variables
8. Return *DFs* and product of remaining factors (expected utility of optimal policy)

Value of Information

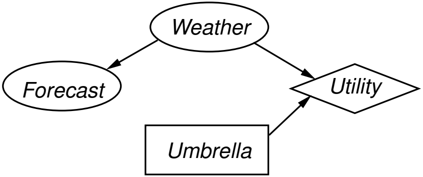
Collecting Information

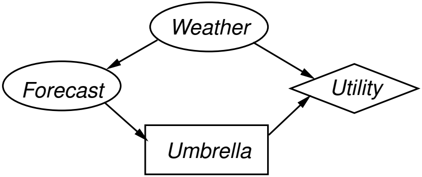
- *CheckSmoke* is aimed at determining (with some uncertainty) the true state of *Smoke* (or even *Fire*)
- *Test* (medical example) is aimed at determining the true state of *Disease*

Value of information

Question: what is it worth to know the exact state of *Forecast* F when making decision *Umbrella*?

Answer: Compare maximal expected utilities of





Collecting Information

- *CheckSmoke* is aimed at determining (with some uncertainty) the true state of *Smoke* (or even *Fire*)
- *Test* (medical example) is aimed at determining the true state of *Disease*

Value of information

Question: what is it worth to know the exact state of a random variable C when making decision D?

Answer: compute

- the expected value val_0 of optimal policy in given decision network
- the expected value val_1 of optimal policy in modified decision network:
  - add an edge from C to D and all subsequent decisions
- $\mathit{val}_1 - \mathit{val}_0$ is the value of knowing C.

Properties of Value of Information

- Value of information is always non-negative
- Value of knowing C for decision D is zero, if no observed value of C can change the decision rule, i.e. for all values $\mathbf{p}$ of existing parents of D, and all values c of C, the optimal decision given $(\mathbf{p}, c)$ is the same as the optimal decision given $(\mathbf{p})$.