Decision Theory

Luigi Portinale

University of Piemonte Orientale, Italy

March 7-10, 2017

1. Decision Problems

2. Decision Networks
   - Influence Diagrams
   - LIMIDs
   - Dynamic Decision Networks

Decision Problems

Decision making

- *Decision*: an irrevocable alllocation of domain resources;
- decisions are taken from observations, in order to maximize some utility (or minimize some cost);
- in real world, the environment of a decision problem is uncertain (decision making under uncertainty)
- *Utility theory*: formalization of preferences and utilities
- *Decision Theory*: combination of probability and utility theories; not claimed to be descriptive, but normative
- *Decision Analysis*: set of practical techniques for applying decision theory to real world situations

Utility theory: preferences

Given two different outcomes A and B:

- $A \succ B$ if we prefer A to B;
- $A \sim B$ if we are indifferent between A and B
- $A \succeq B$ if B is not preferred to A

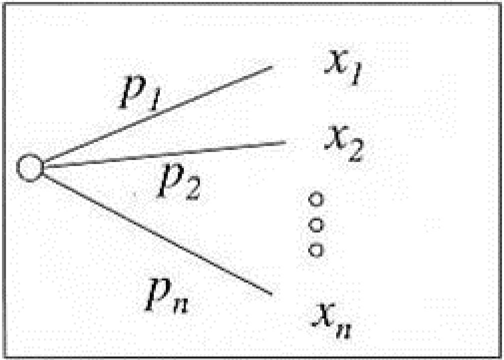
Utility theory: lotteries

A **lottery** is a set of mutually exclusive and exhaustive outcomes, with their associated probabilities of occurrence.

$$L : [S_1 : p_1; \ldots S_n : p_n]$$

where p_i is the probability of occurrence of outcome S_i

A Lottery



Utility theory: rational preferences

Von Neumann-Morgenstein axioms of rationality

- *Completeness:* $(A \succ B) \vee (A \sim B) \vee (A \succeq B)$
- *Transitivity:* $(A \succeq B) \wedge (B \succeq C) \rightarrow (A \succeq C)$
- *Continuity:* if $A \succeq C \succeq B$, then there exists a probability p such that $[A : p; B : 1 - p] \sim C$
- *Independence:* if $A \succ B$, then for any C and probability p, $[A : p; C : 1 - p] \succ [B : p; C : 1 - p]$

The money pump

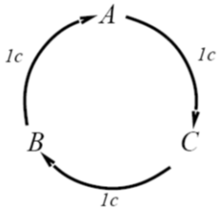
Violating the constraints leads to self-evident irrationality

For example: an agent with intransitive preferences can be induced to give away all its money

If $B \succ C$, then an agent who has C would pay (say) 1 cent to get B

If $A \succ B$, then an agent who has B would pay (say) 1 cent to get A

If $C \succ A$, then an agent who has A would pay (say) 1 cent to get C



Utility Function

Constraints on rational preferences lead to a real-valued utility measure over the space S of possible outcomes

$$U : S \rightarrow \mathbb{R}$$

$U(A) > U(B)$ iff $A \succ B$ and
$U(A) = U(B)$ iff $A \sim B$

Expected Utility

Given a lottery $L : [S_1 : p_1, \ldots S_n : p_n]$ and a utility function U over the space of the outcomes, the lottery *expected utility* is

$$EU(L) = \sum_{i=1}^{n} p_i U(S_i)$$

Preference between lotteries

Given lotteries L_1 and L_2 we say that
$L_1 \succ L_2$ iff $EU(L_1) > EU(L_2)$ and
$L_1 \sim L_2$ iff $EU(L_1) = EU(L_2)$

Lotteries are models of *actions* or *decisions* leading to outcomes.
Given a set of decisions represented as lotteries, the Maximum Expected Utility principle states that the rational choice is to select the one (or one among those) producing the maximum EU.

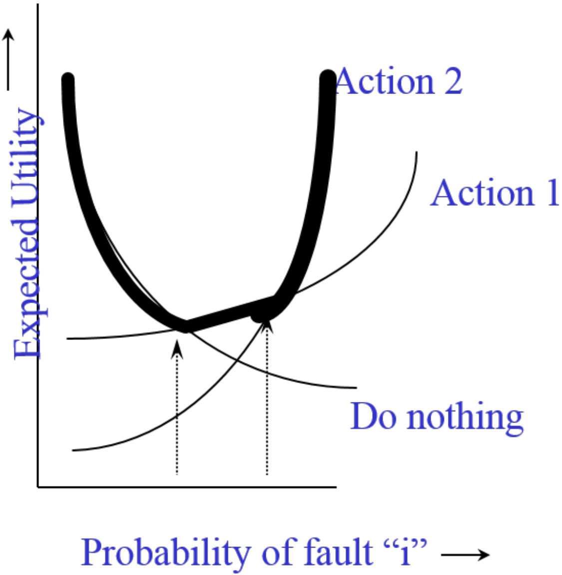
To generalize, suppose we have a probabilistic model providing $P(s'|o,a)$, the probability of obtaining the outcome (state of the world) s' given that we observe o and take action a; the *expected utility* of taking a when observing o is

$$EU(a|o) = \sum_{s'} P(s'|o,a)U(s')$$

MEU Principle

The MEU principle select the action a^* such that

$$a^* = \arg\max_a EU(a|o)$$


Expected Utility
Action 2
Action 1
Do nothing
Probability of fault "i"

Normalized utility

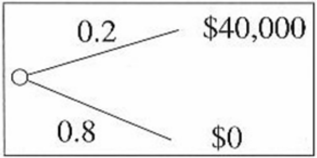
Utility measure can always be normalized such that the worst outcome $S_\perp$ has utility $U(S_\perp) = 0$ and the best outcome $S_\top$ has utility $U(S_\top) = 1$

Utility elicitation

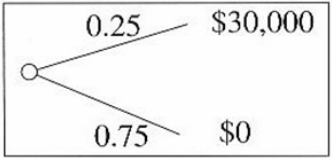
In a normalized utility setting, the utility of an outcome S is $U(S) = p$ where $S \sim [S_\top : p; S_\perp : 1 - p]$

$$U(S) = EU(L) = 1 \times p + 0 \times (1 - p) = p$$

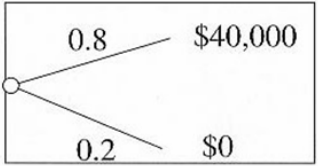
The utility of money



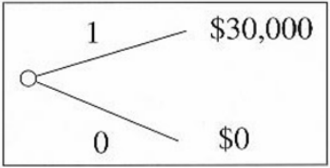
$\succ$



$$0.2 \bullet U(\$40k) > 0.25 \bullet U(\$30k)$$
$$0.8 \bullet U(\$40k) > U(\$30k)$$

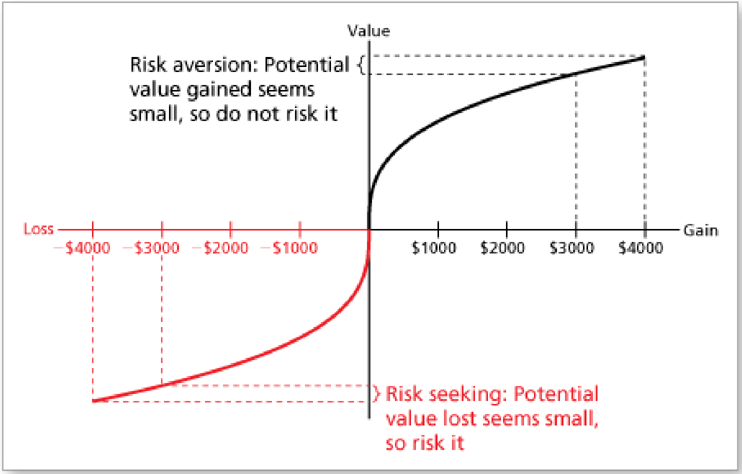


$\prec$



$$0.8 \bullet U(\$40k) < U(\$30k)$$

Decision Theory
Luigi Portinale
Decision Problems
Decision Networks
Influence Diagrams
LIMIDs
Dynamic Decision Networks



Risk Aversion, Neutrality or Propension

- *Risk Neutral*: the utility function is linear; there is no preference between a certain prize and a lottery with expected value equal to the price
- *Risk Aversion*: the utility function is concave (concave down); preference for a certain prize wrt a lottery with expected value equal to the price
- *Risk Seeking*: the utility function is convex (concave up); preference for a lottery with a given prize as expected value wrt the certain prize.

Decision Theory
Luigi Portinale

Decision Problems

Decision Networks
Influence Diagrams
LIMIDs
Dynamic Decision Networks

St. Petersurg Paradox (Bernoulli)

- Toss a fair coin several times: win 2^n money units if *head* come up at toss number n;
- Computing EU of the game in case of linear utility:

$$EU = \sum_{i=1}^{\infty} 2^i \frac{1}{2^i} = 1 + 1 + 1 + \cdots \to \infty$$

- *Paradox*: a player should be willing to pay any sum to play the game!

Decision Theory
Luigi Portinale
Decision Problems
Decision Networks
Influence Diagrams
LIMIDs
Dynamic Decision Networks

St. Petersburg Paradox (solution)

- Computing EU of the game with a concave function (e.g. $\log(x)$):

$$EU = \sum_{i=1}^{\infty} \log(2^i)\frac{1}{2^i} = \sum_{i=1}^{\infty} i\log(2)\frac{1}{2^i} =$$

$$= \log(2)\sum_{i=1}^{\infty} i\frac{1}{2^i} = 2\log(2) = \log(4)$$

- The price to be paid is the inverse function of the utility thus

$$price = e^{\log(4)} \approx 4$$

Decision Networks

Decision Theory
Luigi Portinale
Decision Problems
Decision Networks
Influence Diagrams
LIMIDs
Dynamic Decision Networks

Decision Network

A *Decision Network* is a triple $DN = \langle G, P, U \rangle$ where

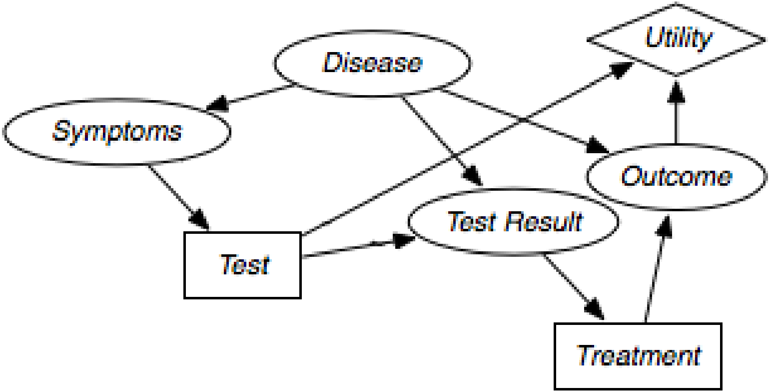
- $G = (V, E)$ is a DAG where:
  - $V = CN \cup DN \cup VN$ with CN *chance nodes* (ovals), DN *decision nodes* (rectangles), VN *value nodes* (diamonds).
  - $E = CA \cup IA \cup FA$ with CA *conditional arcs* entering a chance node, IA *informational arcs* entering a decision node, FA *functional arcs* entering a value node
- P is a parametrized probability distribution

$$Pr(X_1, X_2, \ldots X_n : d_1, \ldots d_m) = \prod_{i=1}^{n} Pr(X_i | \pi(X_i))$$

  with $d_1, \ldots d_m$ assignments to the decision variables
- U is is the joint utility function (additively decomposable) $U = \sum_{v \in VN} U_v$, where $U_v : \Omega_{\pi(v)} \to \mathbb{R}$ is the local utility function of value node v.

Ω_X is the domain of X (i.e., the set of states or values of X)

Example


Utility
Disease
Symptoms
Outcome
Test Result
Test
Treatment

	Disease	no		yes	
	Treatment	no	yes	no	yes
Outcome	bad	0.01	0.01	0.9	0.15
	good	0.99	0.99	0.1	0.85

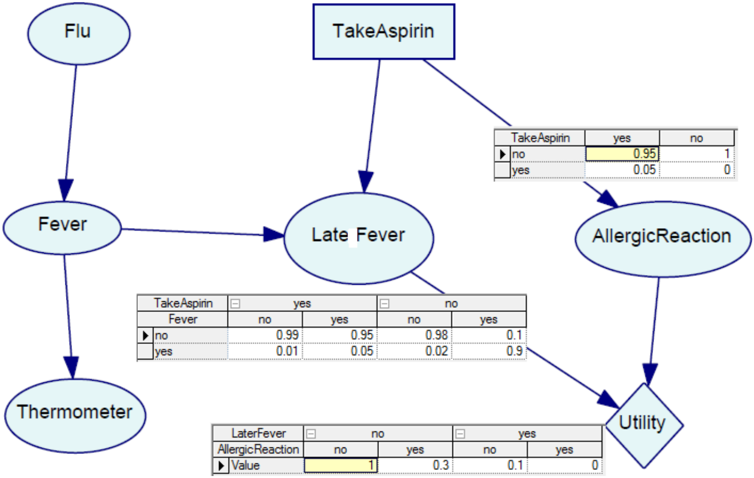
CPT for a chance node (entering conditional arcs)

Test	no		yes	
Outcome	bad	good	bad	good
Utility	0.1	1	0	0.9

Utility function for a value node (entering functional arcs)

To decide whether to perform **Test**, need to observe **Symptoms** (informational arc)

Single Decision, no informational arcs



TakeAspirin	yes	no
no	0.95	1
yes	0.05	0

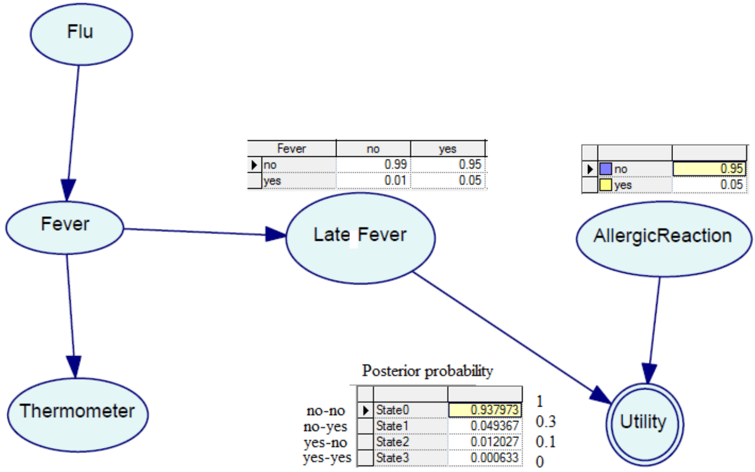
TakeAspirin	yes		no	
Fever	no	yes	no	yes
no	0.99	0.95	0.98	0.1
yes	0.01	0.05	0.02	0.9

LaterFever	no		yes	
AllergicReaction	no	yes	no	yes
Value	1	0.3	0.1	0

Inference Algorithm

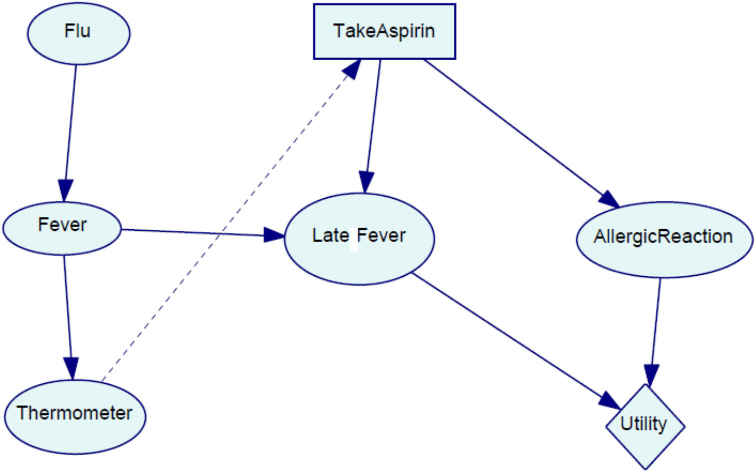
- set evidence (if present)
- for each value of action node
  - compute posterior probability of parents of utility node;
  - compute EU of action value
- return action value with MEU

Decision Theory
Luigi Portinale
Decision Problems
Decision Networks
Influence Diagrams
LIMIDs
Dynamic Decision Networks



$$EU(yes) = 0.938 \times 1 + 0.049 \times 0.3 + 0.012 \times 0.1 + 0.001 \times 0 = 0.954$$

Single Decision, informational arcs

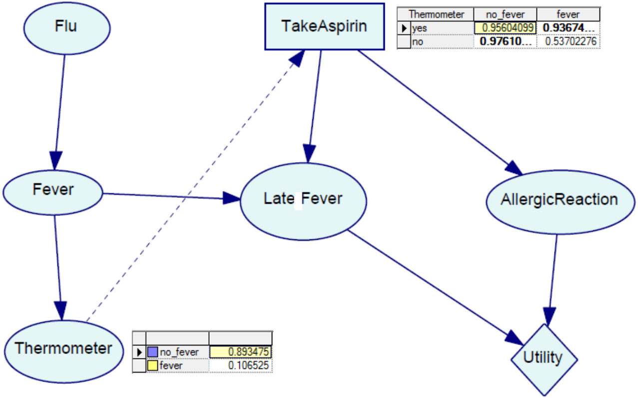


Inference Algorithm

- set evidence (if present)
- for each combination of values of parent nodes of action
    - for each value of action node
        - compute posterior probability of parents of utility node;
        - compute EU of action value
    - return (*combination*, *action*) such that action has MEU

Decision Theory
Luigi Portinale
Decision Problems
Decision Networks
Influence Diagrams
LIMIDs
Dynamic Decision Networks

Single Decision: policy



Optimal policy π:
if measure *no fever* then *don't take aspirin*
if measure *fever* then *take aspirin*
$EU(\pi) = 0.893 \times 0.976 + 0.106 \times 0.937 = 0.971$

Decision Theory
Luigi Portinale
Decision Problems
Decision Networks
Influence Diagrams
LIMIDs
Dynamic Decision Networks

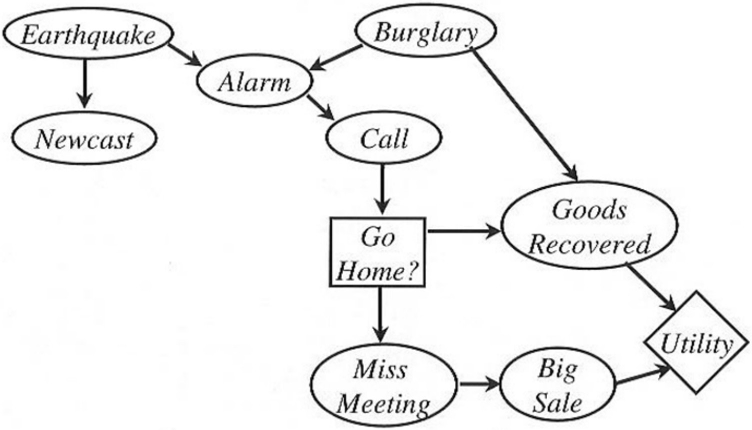
Value of Information (VOI)

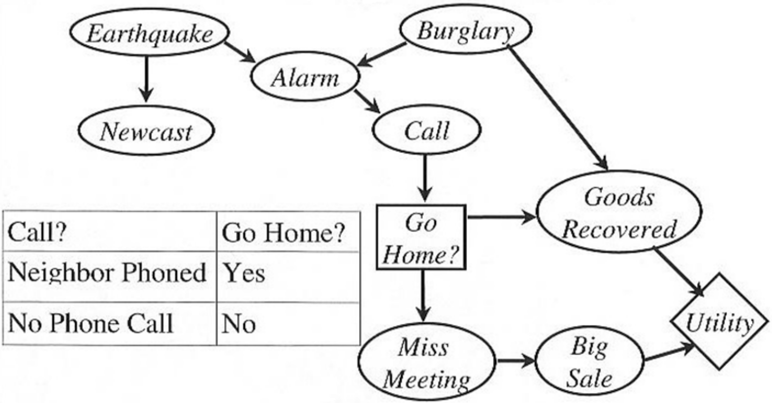
The VOI of a piece of information is the difference between the EU of the best decision taken when the information is available and the EU of the best decision taken when it is not available

Aspirin example

$VOI(\text{Thermometer}) = 0.971 - 0.954 = 0.017$

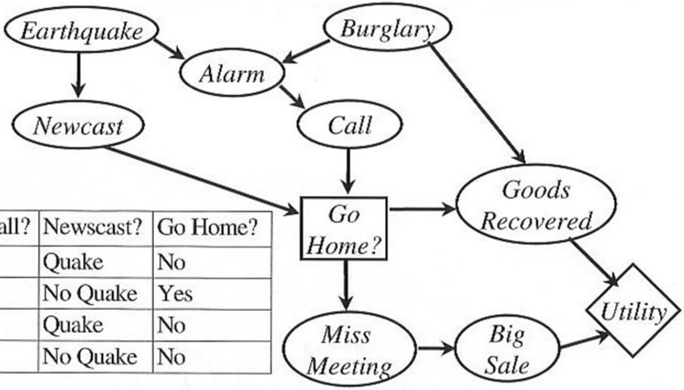
- The VOI is always either positive or null;
- The VOI is the maximum rational price to be paid for gathering the information


Earthquake
Burglary
Alarm
Newcast
Call
Go Home?
Goods Recovered
Utility
Miss Meeting
Big Sale



Call?	Go Home?
Neighbor Phoned	Yes
No Phone Call	No

Expected Utility of this policy is 100



Phonecall?	Newscast?	Go Home?
Yes	Quake	No
Yes	No Quake	Yes
No	Quake	No
No	No Quake	No

Expected Utility of this policy is 112.5

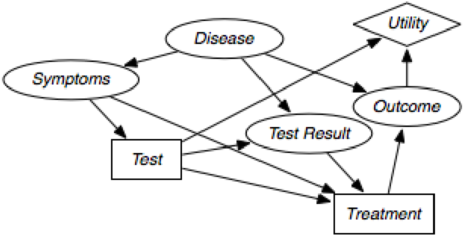
$$VOI(\text{Newcast}) = 12.5$$

Influence Diagrams

Influence Diagram

An *Influence Diagram* (ID) is a decision network with the following properties:

- **Regularity:** decision nodes are connected through a direct path; it follows that decision are temporally ordered
- **No Forgetting:**every decision is conditioned by all previous decisions and related information; this means that any disclosed information (i.e., decisions and observations made) is remembered and considered for future decisions.



Necessary Evidence

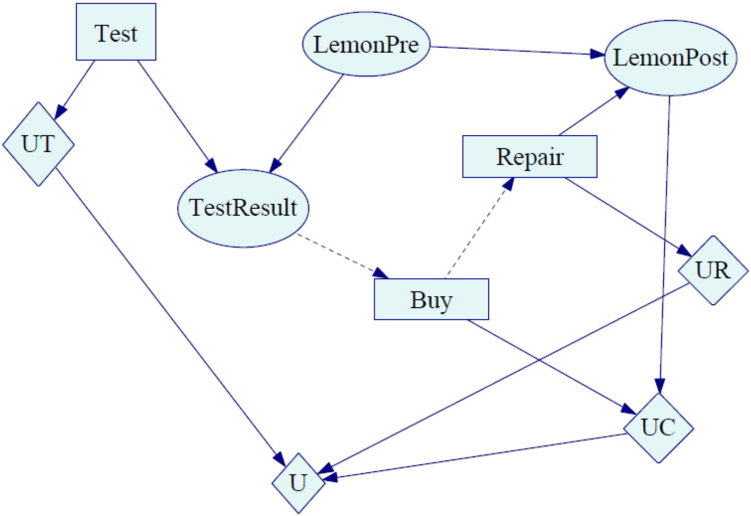
Given a sequence of decisions $\{D_1, \dots D_m\}$, let us define as $E_i = \bigcup_{k=0}^{i-1} E_k \cup \pi(D_{i+1})$ with $E_0 = \pi(D_1)$; we call E_i the *necessary evidence* for D_{i+1}.

ID: Policy

Given an influence diagram and the corresponding sequence of decisions $\{D_1 \dots D_m\}$, for any decision variable $D_i (1 \leq i \leq m)$, a policy δ_{D_i} is a function specifying an instance of D_i for any configuration of its necessary evidence, that is $\delta_{D_i} : \Omega_{E_{i-1}} \rightarrow \Omega_{D_i}$. If $E_0 = \emptyset$ (i.e., the first decision D_1 has no parents), then $\delta_{D_1} \in \Omega_{D_1}$ (i.e., it is simply a valid state of D_1)

The car buyer problem

- I need a car and my colleague John can sell me a used car (market value €$12K$) for €$10K$
- I don't totally trust John; he can sell me a *lemon* (defective car) with probability 20%
- I can test for the defect (cost €25); the test is completely reliable if the car is a *peach* (good car), while it can discover a lemon 99% of the times;
- After deciding about the test, I can choose whether to buy the car, keeping in mind that I can possibly repair the car (cost €150); repair can trasform a lemon into a peach 95% of the times;
- Suppose utilities are modeled as additive negative costs.


Test
LemonPre
LemonPost
UT
Repair
TestResult
UR
Buy
UC
U

- analyze the best sequence of decisions
- what changes if test reports a lemon or a peach?
- what changes if repair is 10 times more expensive?

LIMIDs

Limited Memory Influence Diagrams

A LIMID is a decision network where regularity and no-forgetting assumptions are removed

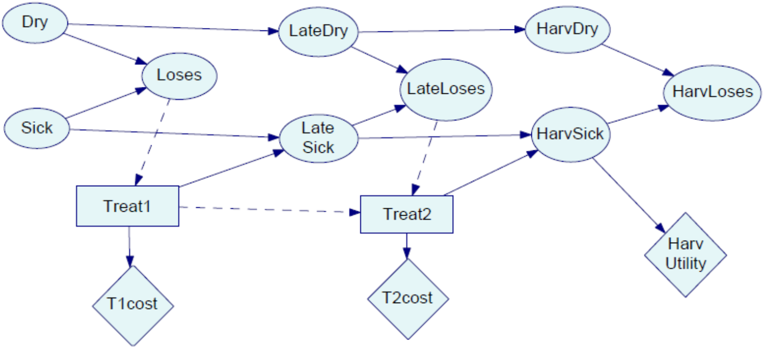
LIMID: Policy

Given a LIMID, for any decision variable $D \in DN$, a *policy* δ_D is a function specifying an instance of D for any configuration of its parent variables, that is $\delta_D : \Omega_{\pi(D)} \rightarrow \Omega_D$. If $\pi(D) = \emptyset$ (i.e., D has no parents), then $\delta_D \in \Omega_D$ (i.e., it is simply a valid state of D).

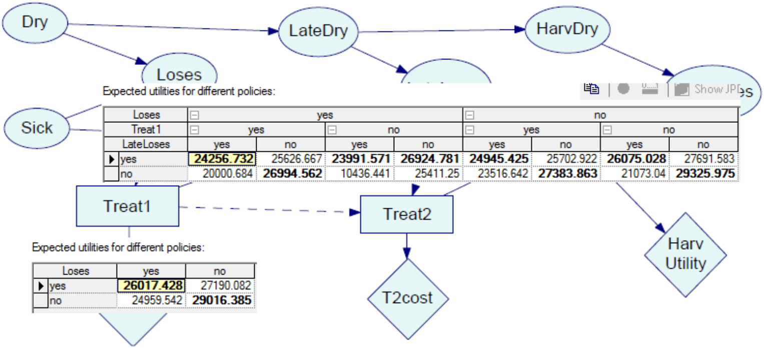
Strategy

Given a decision network (either an ID or a LIMID), a *strategy* is a set $q = \{\delta_D : D \in DN\}$ of policies for the decisions.

An Influence Diagram



Influence Diagram: policies

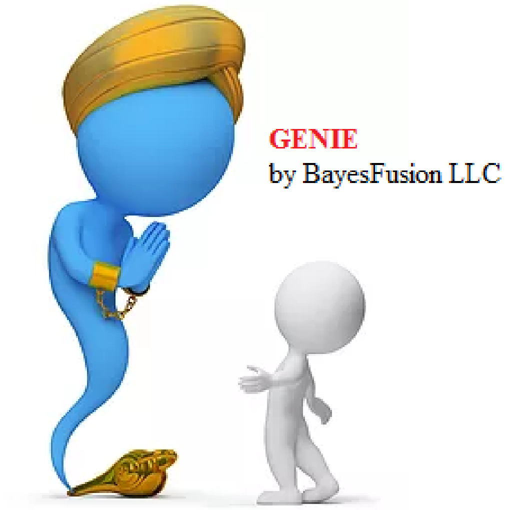


Expected utilities for different policies:

Loses	yes				no			
Treat1	yes		no		yes		no	
LateLoses	yes	no	yes	no	yes	no	yes	no
yes	**24256.732**	25626.667	**23991.571**	**26924.781**	**24945.425**	25702.922	**26075.028**	27691.583
no	20000.684	**26994.562**	10436.441	25411.25	23516.642	**27383.863**	21073.04	**29325.975**

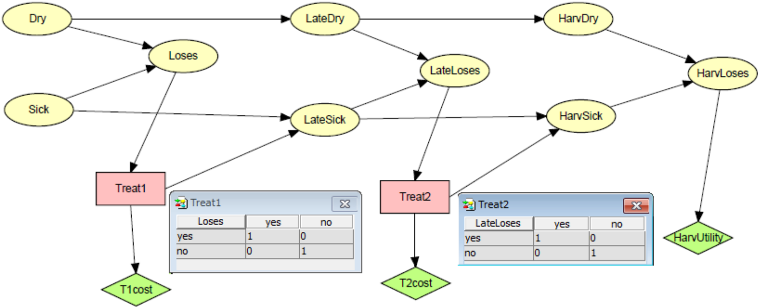
Expected utilities for different policies:

Loses	yes	no
yes	**26017.428**	27190.082
no	24959.542	**29016.385**



http://www.bayesfusion.com

A LIMID with Policies



Treat1

Loses	yes	no
yes	1	0
no	0	1

Treat2

LateLoses	yes	no
yes	1	0
no	0	1

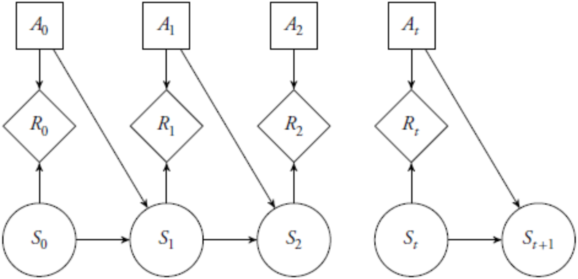
Decision Theory
Luigi Portinale
Decision Problems
Decision Networks
Influence Diagrams
LIMIDs
Dynamic Decision Networks

HUGINEXPERT

http://www.hugin.com

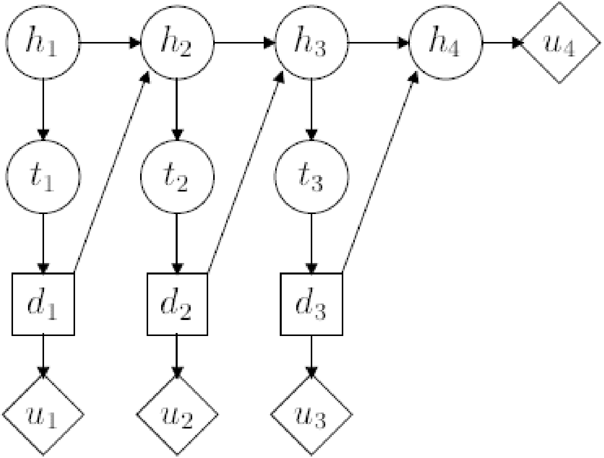
Dynamic Decision Networks

- By assuming a stationary model we can consider only adjacent time slices or a plate representation
- For a finite horizon, the unrolling produces the desired model

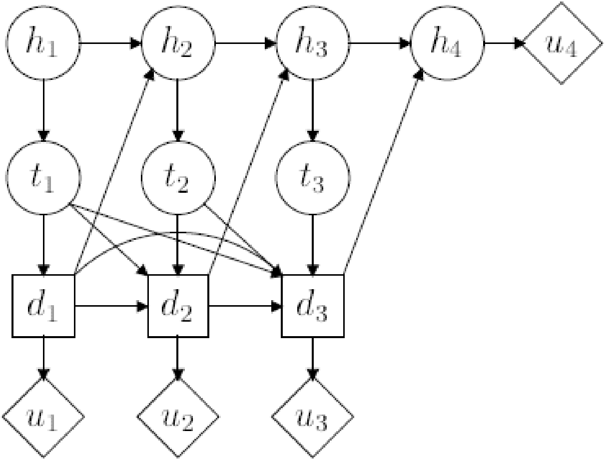


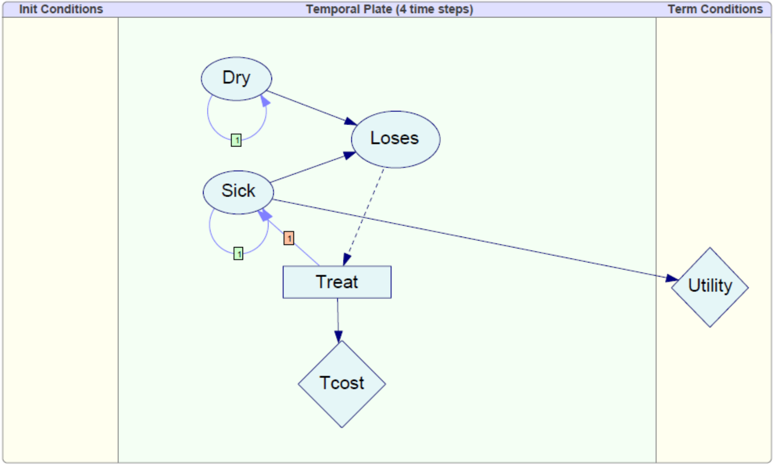
(a) General representation. (b) Stationary representation.

Interpreted as a LIMID

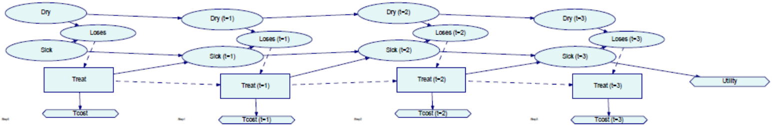


Interpreted as an ID




Init Conditions
Temporal Plate (4 time steps)
Term Conditions
Dry
Loses
Sick
Treat
Tcost
Utility
1
1
1

Influence Diagram



Treat(t=0)

Loses	Si	No
Si	25994.732	26979.586
No	**26431.264**	**28829.202**

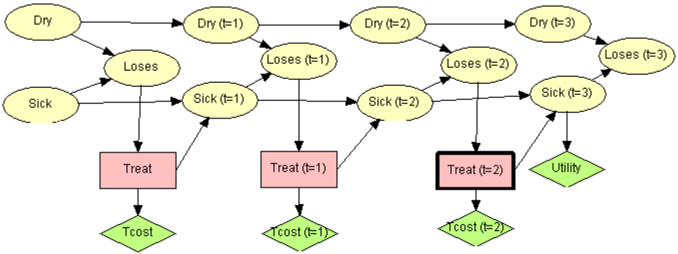
Treat(t=1)

Loses	Si				No			
Treat	Si		No		Si		No	
Loses (t=1)	Si	No	Si	No	Si	No	Si	No
Si	24207.791	25143.695	**25764.194**	26883.688	24496.03	25213.712	**26304.036**	27210.135
No	**24280.267**	**26946.209**	23992.924	**27785.597**	**25432.117**	**27113.179**	26123.081	**29094.986**

Treat(t=2)

Loses	Si								
Treat	Si								
Loses (t=1)	Si				No				
Treat (t=1)	Si		No		Si		No		
Loses (t=2)	Si	No	Si	No	Si	No	Si	No	
Si	**23030.077**	23655.865	**23446.572**	25349.658	**23092.088**	23702.434	**24288.142**	25689.211	**2374**
No	21948.813	**25143.626**	15864.603	**25580.358**	22265.397	**25381.373**	20161.039	**27313.864**	174

LIMID



Treat

Loses	Si	No
Si	0	0
No	1	1

Treat (t=1)

Loses (t=1)	Si	No
Si	1	0
No	0	1

Treat (t=2)

Loses (t=2)	Si	No
Si	1	0
No	0	1

Decision Theory
Luigi Portinale
Decision Problems
Decision Networks
Influence Diagrams
LIMIDs
Dynamic Decision Networks

Pure and Random Policies

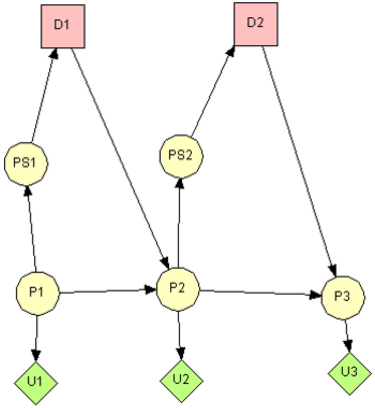
- A *pure policy* is a mapping from states of parent nodes of a decision node, to a state of the decision node
- A *random policy* is a probability distribution over the states of the decision node, given the state of the parents
- A pure policy is a special case of random policy

Let δ_d be a policy (either pure or random) for decision d and $q = \{\delta_d : d \in DN\}$ be a strategy;
The probability of a strategy q is given by

$$f_q = \prod_r p_r \prod_d \delta_d$$

where p_r are the local distributions over the random variables r and δ_d is either 0 or 1 in case of pure policy.

Example: robot positioning



- P_i robot position at time i
- PS_i measured (by sensor) robot position at time i
- $D_i = \{l, r, s\}$ movement action (*left*, *right*, *stay*)
- U_i utility of robot position at time i (1 for center, 0 otherwise)

Pure policies:

$$\delta_1 : \{PS_1 = l \to D_1 = r; PS_1 = c \to D_1 = s; PS_1 = r \to D_1 = l\}$$
$$\delta_2 : \{PS_2 = l \to D_2 = r; PS_2 = c \to D_2 = s; PS_2 = r \to D_2 = l\}$$

$$P(D_1 = l|PS_1 = l) = 0; P(D_2 = l|PS_2 = l) = 0; \ldots$$
$$P(D_1 = r|PS_1 = l) = 1; P(D_2 = s|PS_2 = c) = 1; \ldots$$

Let $q = \{\delta_1, \delta_2\}$, we compute $f_q(P_1, PS_1, D_1, P_2, PS_2, D_2, P_3)$:

$$f_q(l, l, l, l, l, l, l) = P(P_1 = l)P(PS_1 = l|P1 = l)P(P_2 = l|P_1 = l, D_1 = l)P(PS_2 = l|P_2 = l)P(P_3 = l|P_2 = l, D_2 = l)\times P(D_1 = l|PS_1 = l)P((D_2 = l|PS_2 = l)$$

$$f_q(l, l, r, c, c, s, c) = P(P_1 = l)P(PS_1 = l|P1 = l)P(P_2 = c|P_1 = l, D_1 = r)P(PS_2 = c|P_2 = c)P(P_3 = c|P_2 = c, D_2 = s)\times P(D_1 = r|PS_1 = l)P((D_2 = s|PS_2 = c)$$

...

$$EU(q) = \sum_x f_q(x)U(x) = \sum_x f_q(x) \sum_u U_u(\pi(u))$$

where u are the utility nodes.

- A *global maximum strategy* $\hat{q}$ is such that $\forall q \; EU(\hat{q}) \geq EU(q)$
- let $\delta'_{d_0} * q = \{\delta'_{d_0}\} \cup q_{-d_0}$ denote the strategy obtained from q by replacing δ_{d_0} with δ'_{d_0}
- A *locally maximum policy* for strategy q at d is a policy $\tilde{\delta}_d$ such that $EU(\tilde{\delta}_d * q) = \sup_{\delta'_d} EU(\delta'_d * q)$
- A strategy $\tilde{q}$ is a *local maximum strategy* if all its policies are local maximum policies at the corresponding decisions i.e., $\forall d, \delta_d$ we have $EU(\tilde{q}) \geq EU(\delta_d * \tilde{q})$ (the EU does not increase by changing only one of its policies)

Algorithm Single Policy Updating

$q \leftarrow$ random select initial strategy
repeat
  $PrevEU \leftarrow EU(q)$
  for all $\delta_d \in q$ **do**
    find local max policy δ'_d
    $q \leftarrow \delta'_d * q$
  end for
until $EU(q) = PrevEU$

Single Policy Updating

- Finding local maximum policy step can be implemented through junction tree inference [Lauritzen & Nilsson 2001]
- There is always a pure local maximum policy (but it may not be unique)
- SPU is an iterative improvment algorithm; the algorithm usually finds the globally optimal policies, but it is possible that the algorithm may get stuck at a local maximum.

Inference in LIMID

Unless all relevant information has been specified as parents, then it can be useful to recompute policies whenever new information becomes available. This is because the computations take all existing observations (in addition to future observations specified as parents of decision nodes) into account when policies are computed.

Suppose in the Apple tree example we have the evidence *Loses* = *yes* ($t = 0$) and we choose *Do not treat*, then we observe *Loses* = *yes* ($t = 1$) and we choose again *Do not treat*, finally we observe *Loses* = *no* ($t = 2$); by considering the policy for treatment at time $t = 2$ we should decide *Do not treat*, resulting in $EU = 21541$; however by choosing *Treat* we get $EU = 26166$. SPU must be run with the new evidence to get this result; in ID it is pre-computed

Loses	Si						
Treat			⊟ No				
Loses (t=1)	o		⊟ Si				⊟
Treat (t=1)	⊟ No		⊟ Si		⊟ No		⊟
Loses (t=2)	Si	No	Si	No	Si	No	Si
Si	**24288.142**	25689.211	**23747.633**	25600.02	**23357.228**	**26166.814**	**24456.1**
No	20161.039	**27313.864**	17401.601	**26858.525**	7197.9534	21541.629	21018.8