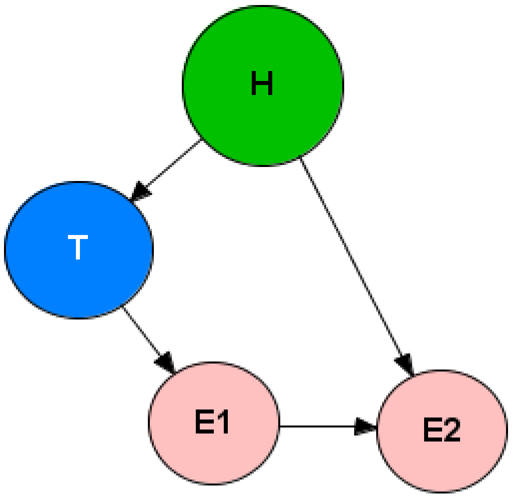
Meeting 10:
Using Bayesian (decision) networks

Bayesian network (BayesNet, BN)

A connected directed acyclic graph (DAG) in which



• the nodes (vertices) represent *random variables*

• the links (edges, arcs) represent direct *relevance* relationships among variables

• The probability distribution of a node satisfies the local Markov property: The conditional probability distribution of a node given the states of its *parent* nodes does not depend on the states of its *descendant* nodes.

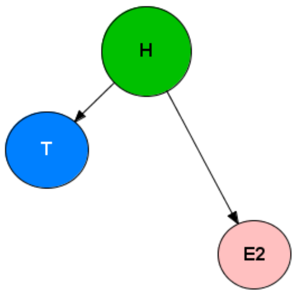
• The probability density (mass) function of the joint distribution of a network with n nodes is

$$f(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} f\left(x_i \middle| \boldsymbol{x}_{\mathrm{PA}(x_i)}\right)$$

where $\mathrm{PA}(x_i)$ is the set of parent nodes of node i

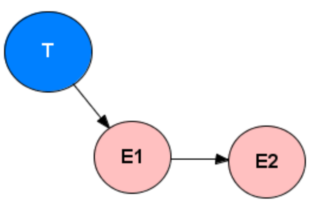
$$f(h, t, e1, e2) = f(h) \cdot f(t|h) \cdot f(e1|t) \cdot f(e2|h, e1)$$

Three connections



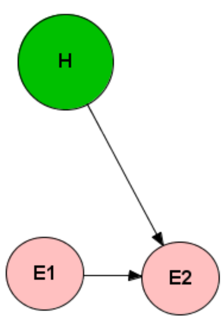
Divergent connection

T and E2 are conditionally independent given H



Serial connection

T and E2 are conditionally independent given E1



Convergent connection

H and E1 are conditionally *dependent* given E2

Completion of the BN: Probability "tables"

Each node (random variable) has *either* discrete states (nominal or ordinal scale) with a probability mass function *or* continuous states with a probability density function.

A probability density function can be sampled into a finite set of states and approximated by a probability mass function over these states (most software for Bayesian network modelling has no "engine" to handle arbitrary continuous distributions).

For a node that is *solely* a parent node:

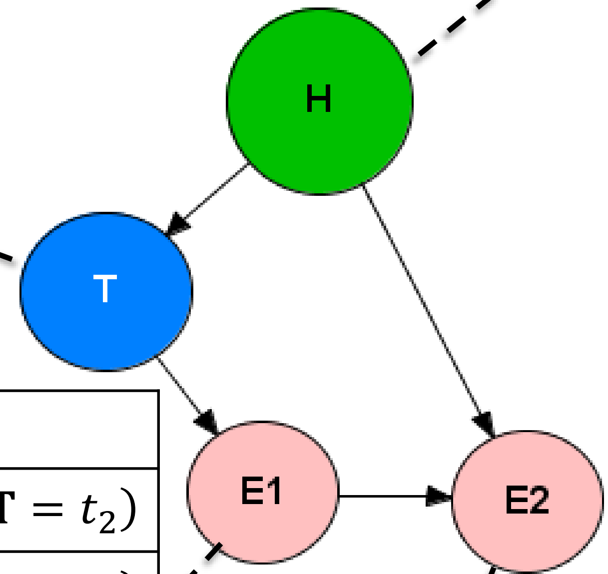
The assigned probabilities (or density function) are conditional on background information only (may be expressed as unconditional or *prior* probabilities)

For a node that is a child node (solely or joint parent/child):

The assigned probabilities (or density function) are conditional on the states of its parent nodes (and on background information).

Example: Two states in each node

H:		
	h_1	$P(\mathbf{H} = h_1)$
	h_2	$P(\mathbf{H} = h_2)$



	H:	h_1	h_2
T:	t_1	$P(\mathbf{T} = t_1 \vert \mathbf{H} = h_1)$	$P(\mathbf{T} = t_1 \vert \mathbf{H} = h_2)$
	t_2	$P(\mathbf{T} = t_2 \vert \mathbf{H} = h_1)$	$P(\mathbf{T} = t_2 \vert \mathbf{H} = h_2)$

	T:	t_1	t_2
E1:	e_{11}	$P(\mathbf{E1} = e_{11} \vert \mathbf{T} = t_1)$	$P(\mathbf{E1} = e_{11} \vert \mathbf{T} = t_2)$
	e_{12}	$P(\mathbf{E1} = e_{12} \vert \mathbf{T} = t_1)$	$P(\mathbf{E1} = e_{12} \vert \mathbf{T} = t_2)$

	H:	h_1		h_2	
	E1:	e_{11}	e_{12}	e_{11}	e_{12}
E2:	e_{21}	$P\left(\mathbf{E2} = e_{21} \middle\vert \begin{matrix}\mathbf{H} = h_1, \\ \mathbf{E1} = e_{11}\end{matrix}\right)$	$P\left(\mathbf{E2} = e_{21} \middle\vert \begin{matrix}\mathbf{H} = h_1, \\ \mathbf{E1} = e_{12}\end{matrix}\right)$	$P\left(\mathbf{E2} = e_{21} \middle\vert \begin{matrix}\mathbf{H} = h_2, \\ \mathbf{E1} = e_{11}\end{matrix}\right)$	$P\left(\mathbf{E2} = e_{21} \middle\vert \begin{matrix}\mathbf{H} = h_2, \\ \mathbf{E1} = e_{12}\end{matrix}\right)$
	e_{22}	$P\left(\mathbf{E2} = e_{22} \middle\vert \begin{matrix}\mathbf{H} = h_1, \\ \mathbf{E1} = e_{11}\end{matrix}\right)$	$P\left(\mathbf{E2} = e_{22} \middle\vert \begin{matrix}\mathbf{H} = h_1, \\ \mathbf{E1} = e_{12}\end{matrix}\right)$	$P\left(\mathbf{E2} = e_{22} \middle\vert \begin{matrix}\mathbf{H} = h_2, \\ \mathbf{E1} = e_{11}\end{matrix}\right)$	$P\left(\mathbf{E2} = e_{22} \middle\vert \begin{matrix}\mathbf{H} = h_2, \\ \mathbf{E1} = e_{12}\end{matrix}\right)$

- For a *statistician*, BN:s are models to be used for making inference and/or classification/prediction.

- However, using BN software with graphical user interfaces, BN:s can be a good tool to explain a statistical model to a *practitioner.*

- Practitioners outside the fields of statistics, computer science, econometrics, theoretical physics, theoretical biology, … <u>tend to</u>

    - be reluctant to the use of mathematical formulas.
    - be reluctant to computer programming and algorithms.

- *Communication skills* are very important for a statistician working in a multi-scientific environment.

Example

Return to the example with banknotes.

Let H_0: Dye is present

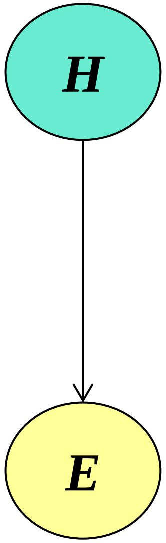
H_1: Dye is not present

and let E_1: Method gives positive detection

E_2: Method gives negative detection

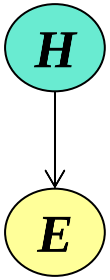
Method of detection gives a positive result (detection) in 99 % of the cases when the dye is present, i.e. the proportion of false negatives is 1% and a negative result in 98 % of the cases when the dye is absent, i.e. the proportion of false positives is 2%

The presence of dye is rare: prevalence is about 0.1 %



H	H_0	0.001
	H_1	0.999

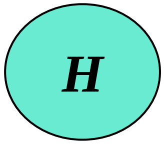
	H:	H_0	H_1
E	E_1	0.99	0.02
	E_2	0.01	0.98



Now, the model has to be executed (run).

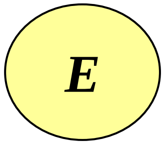
This means that the marginal distributions in each node are calculated

For node ***H*** , the marginal distribution is the (prior) probabilities entered in the probability table (since this node has no parents).



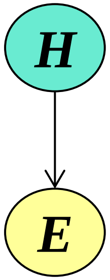
H	H_0	0.001
	H_1	0.999

For node ***E*** , the marginal distribution is calculated using the law of total probability:



E	E_1	$P(\boldsymbol{E} = E_1) = P(\boldsymbol{E} = E_1 \mid \boldsymbol{H} = H_0) \cdot P(H = H_0)$ $+ P(\boldsymbol{E} = E_1 \mid \boldsymbol{H} = H_1) \cdot P(H = H_1) = 0.99 \cdot 0.001 + 0.02 \cdot 0.999$ $= 0.02097$
	E_2	$1 - 0.02097 = 0.97903$

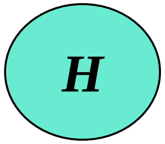
This is what the “engine” (algorithm) in BN software does – applying probability calculus.



In an executed mode, different inferences may be done by fixing (instantiating) the state of one or several nodes.

Here, the instantiation is entering the data: $\boldsymbol{E} \equiv E_1$

Consequently, the probabilities in node $\boldsymbol{H}$ are updated:



E		E_1
H	H_0	$P(H_0 \vert E_1) = \frac{P(E_1 \vert H_0) \cdot P(H_0)}{P(E_1 \vert H_0) \cdot P(H_0) + P(E_1 \vert H_1) \cdot P(H_1)}$ $= \frac{0.99 \cdot 0.001}{0.99 \cdot 0001 + 0.02 \cdot 0.999} \approx 0.047$
	H_1	$\approx 1 - 0.047 = 0.953$

Again, this is what the engine does.

Example Who smashed the window?

A window (pane) was smashed and a person, Mr G is suspected for having done it. On Mr G's pullover 8 glass fragments were recovered, they all matched the (pane of) the smashed window.

Let

H be a random variable with states
H_1= "Mr G smashed the window" and
H_2: "Someone (or something) else smashed the window".

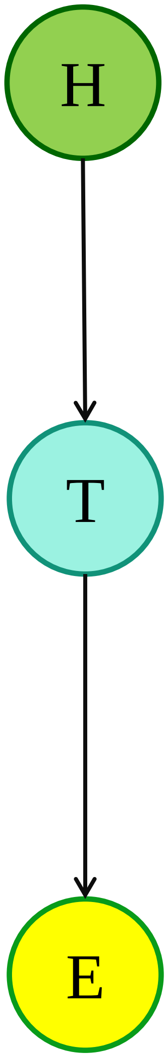
T be a random variable for which the states are the number of fragments transferred to Mr G's pullover when the window was smashed. Note that if Mr G's pullover was not sufficiently near the window when it was smashed, then $T = 0$.

E be a random variable for which the states are the number of fragments that could be (and were) recovered from Mr G's pullover. Note that *E* is not equal to *T* since (i) it cannot be assumed that all fragments transferred to Mr G's pullover persisted and (ii) were detectable when analysing it.

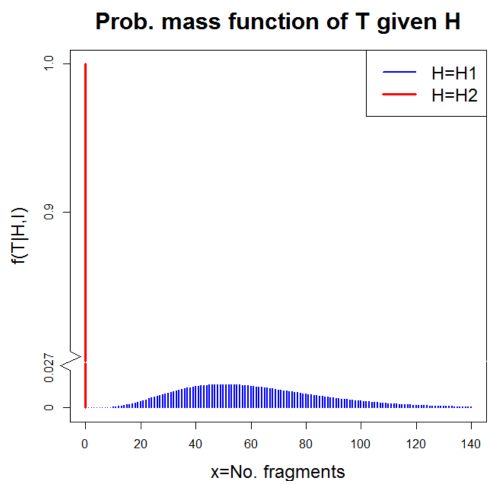
Serial connection

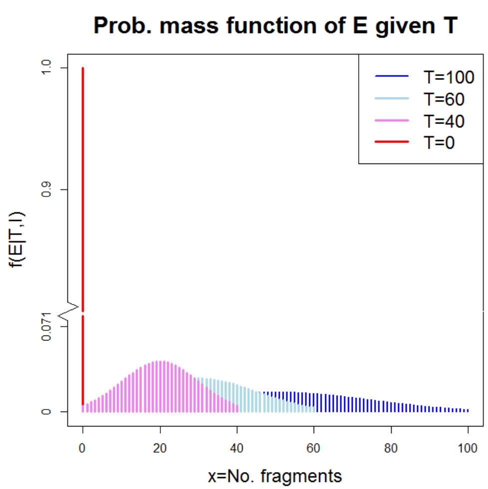
H_1= "Mr G smashed the window"

H_2: "Someone (or something) else smashed the window".



H	Probability
H_1	$P(\mathrm{H} = H_1 \mid I)$
H_2	$P(\mathrm{H} = H_2 \mid I)$



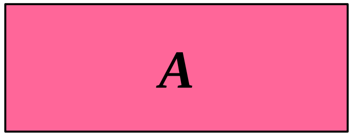


Once the value of **T** is known the state of **H** is no longer relevant for the state of **E**.

Influence diagrams

Decision-theoretic components can be added to a Bayesian network. The complete network is then related to as a *Bayesian Decision Network* or more common ***Influence diagram* (*ID*)**

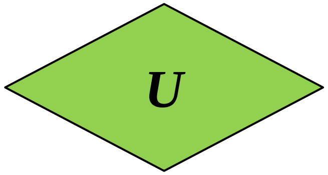
Two additional nodes are used



Action (or *Decision*) **node**

All (*m*) actions are specified in a “table”

A
Action 1
Action 2
⋮
Action *m*



Utility (or *Loss*) **node**

All consequences (measured as utilities or losses) are specified.

Must be the child node of the action node and the node with the states of the world.

The decision matrix

	s_1	$\cdots$	s_n
a_1	U_{11}	$\cdots$	U_{1n}
$\vdots$	$\vdots$	$\ddots$	$\vdots$
a_m	U_{m1}	$\cdots$	U_{mn}

is restructured to a "BN table":

Action:	a_1			$\cdots$	a_m		
State:	s_1	$\cdots$	s_n	$\cdots$	s_1	$\cdots$	s_n
Utility:	U_{11}	$\cdots$	U_{1n}	$\cdots$	U_{m1}	$\cdots$	U_{mn}

Example: Dye on banknotes

- The banknote is a SEK 100 banknote.
- If we deem the banknote to have been contaminated with the dye, we will consider it as useless, and it will be destroyed.
- If we deem the banknote not to have been contaminated with the dye, we will use it (in the future) for ordinary purchasing.
- Upon using the banknote for purchasing, if it is revealed (by other means than our method) that the banknote is contaminated with the dye, there is a fine of SEK 5000. ***Assume that if it is contaminated this will be revealed!***

Hence, a payoff function for this problem is

Action	State of the world	
	Dye is present (H_0)	Dye is not present (H_1)
Destroy banknote	0	–100
Use banknote	–5000	0

Note that the amounts of money should be entered as negative payoffs. If our utilities are linear in money, this is also our (dis)utility function

We may however consider a loss function to better describe the situation.

Recall: $L(a, \boldsymbol{\theta}) = \max_{a' \in \mathcal{A}} \left(U(a', \boldsymbol{\theta}) \right) - U(a, \boldsymbol{\theta})$

Action	State of the world	
	Dye is present (H_0)	Dye is not present (H_1)
Destroy banknote	0 – 0 = **0**	0 – (–100) = **100**
Use banknote	0 – (–5000) = **5000**	0 – 0 = **0**

but is this description so much better than the one with (dis)utilities?

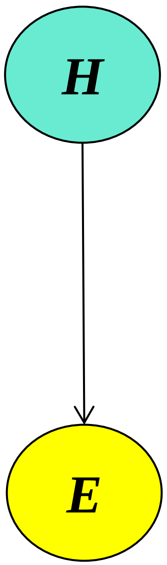
Using the Bayesian network constructed before:

H_0: Dye is present

H_1: Dye is not present

E_1: Method gives positive detection

E_2: Method gives negative detection

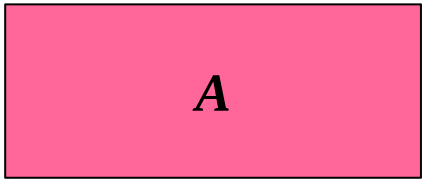


H	H_0	0.001
	H_1	0.999

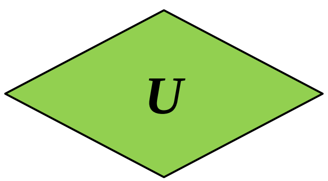
	H:	H_0	H_1
E	E_1	0.99	0.02
	E_2	0.01	0.98

This is the *inference* part of the network model.

Now, we add one node for the actions that can be taken and one for the utility function



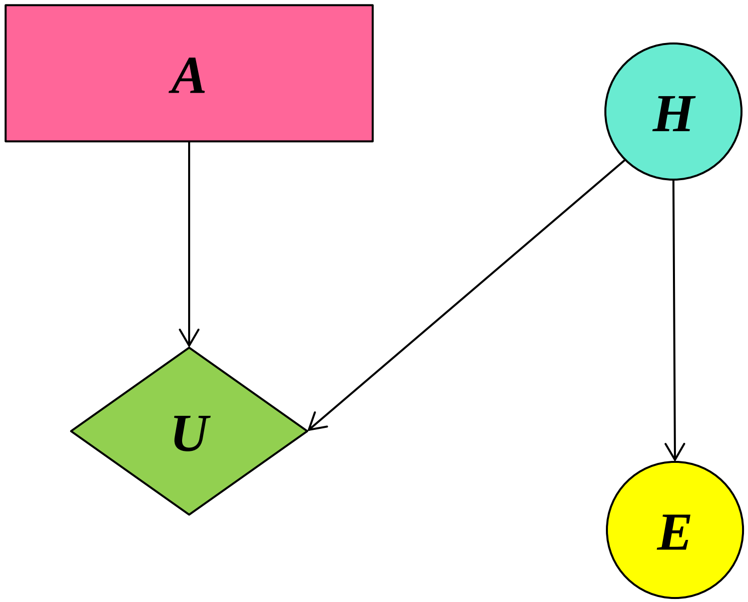
A	
a_1	*Destroy banknote*
a_2	*Use banknote*



A:	a_1		a_2	
H:	H_0	H_1	H_0	H_1
U:	0	–100	–5000	0

This is the *decision* part of the network model.

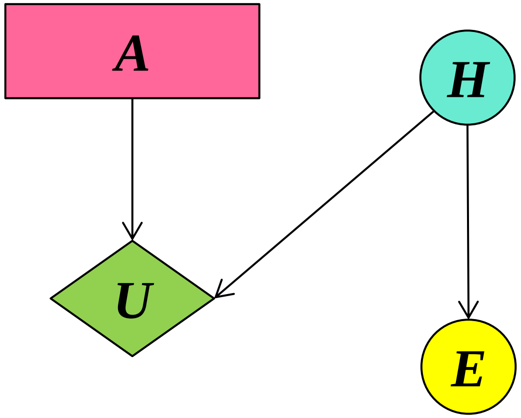
The influence diagram



With this network we would like to be able to propagate data (backwards) from node ***E*** to a choice of decision in node ***A***.

Hence, in node ***A*** the posterior expected utility should be calculated, and the utilities should be specified in node ***U***.

When the influence diagram is executed:



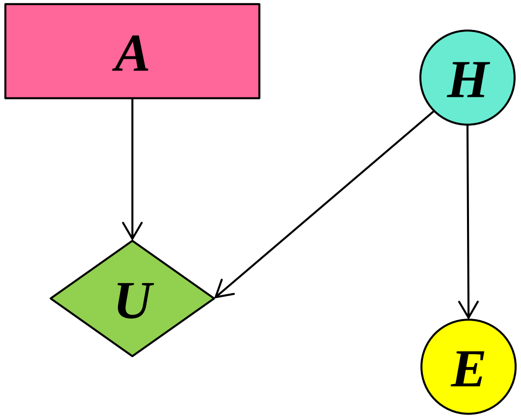
- Probabilities in node ***H*** are not affected

H	H_0	0.001
	H_1	0.999

- Prior expected utilities are calculated in node ***A*** from the probabilities in node ***H*** and the utilities in node ***U***

A	
a_1	$E\big(U(a_1)\big) = 0 \cdot 0.001 + (-100) \cdot 0.999 = -99.9$
a_2	$E\big(U(a_2)\big) = (-5000) \cdot 0.001 + 0 \cdot 0.999 = -5$

Hence, the optimal action <u>in prior sense</u> is a_2 = “Use banknote”



In the executed model, inference is now made by instantiating node $\boldsymbol{E}$ to state E_1, i.e. entering the data: "Method gives positive detection"

- Probabilities in node $\boldsymbol{H}$ are updated

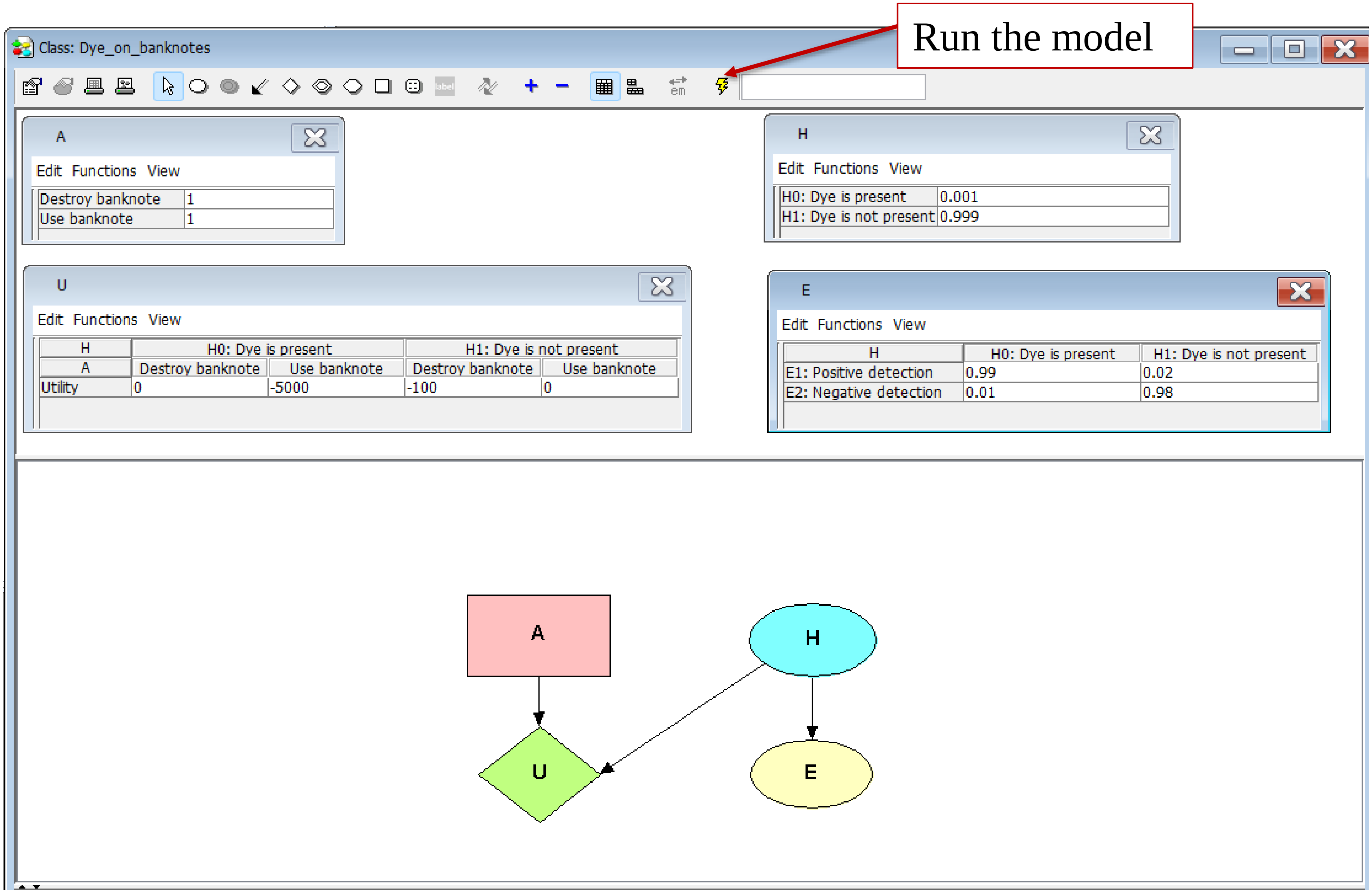
$\boldsymbol{H}$	H_0	0.047
	H_1	0.953

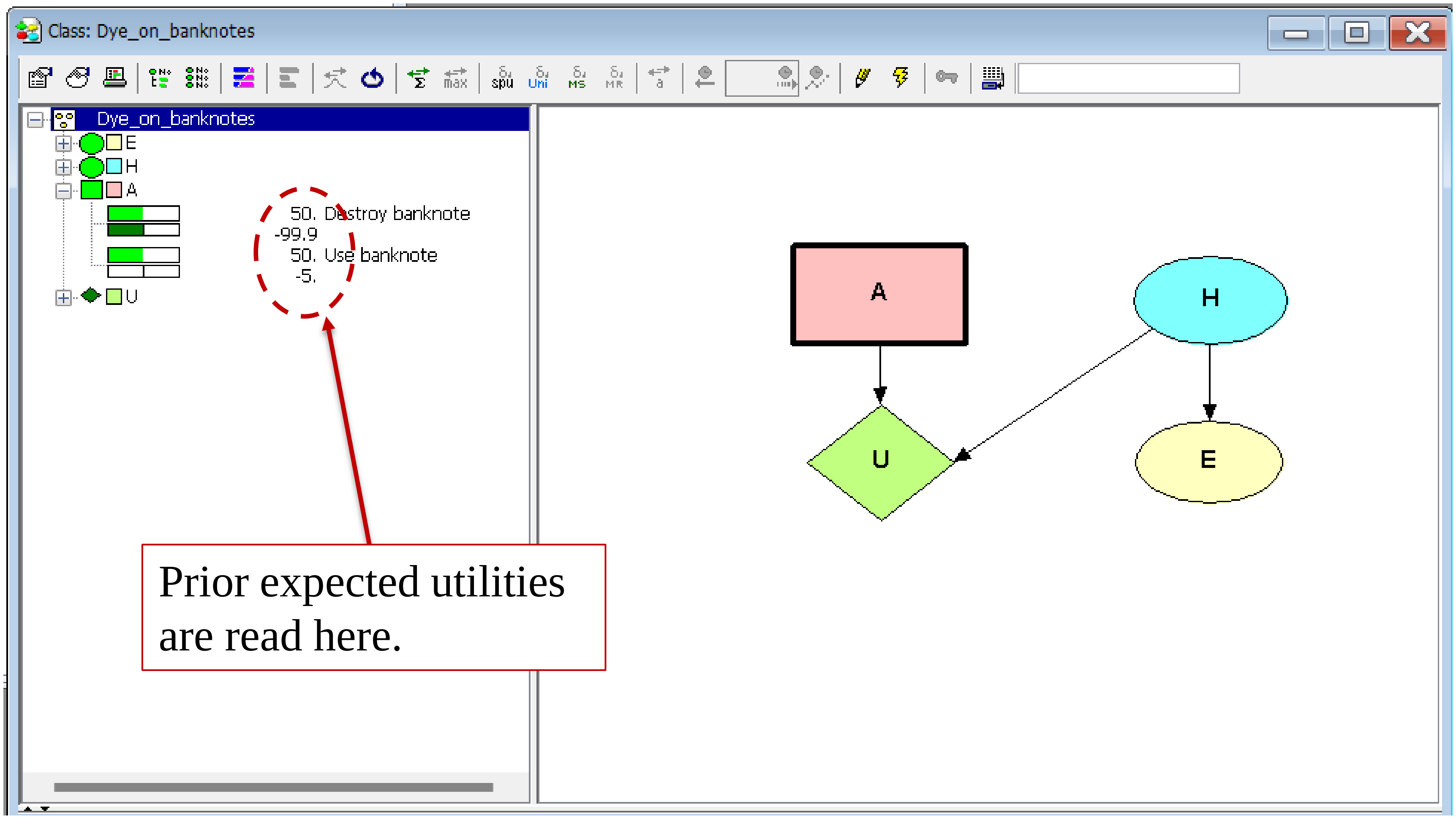
- Posterior expected utilities are calculated in node $\boldsymbol{A}$ from the updated probabilities in node $\boldsymbol{H}$ and the utilities in node $\boldsymbol{U}$

$\boldsymbol{A}$	
a_1	$E(U(a_1)\mid\boldsymbol{E}=E_1)=0\cdot 0.047+(-100)\cdot 0.953\approx -95.3$
a_2	$E(U(a_2)\mid\boldsymbol{E}=E_1)=(-5000)\cdot 0.047+0\cdot 0.953\approx -235.0$

Hence, the optimal action <u>in posterior sense</u> is a_1 = "Destroy banknote"

Using software (exemplifying with Hugin Lite® from Hugin Expert A/S)

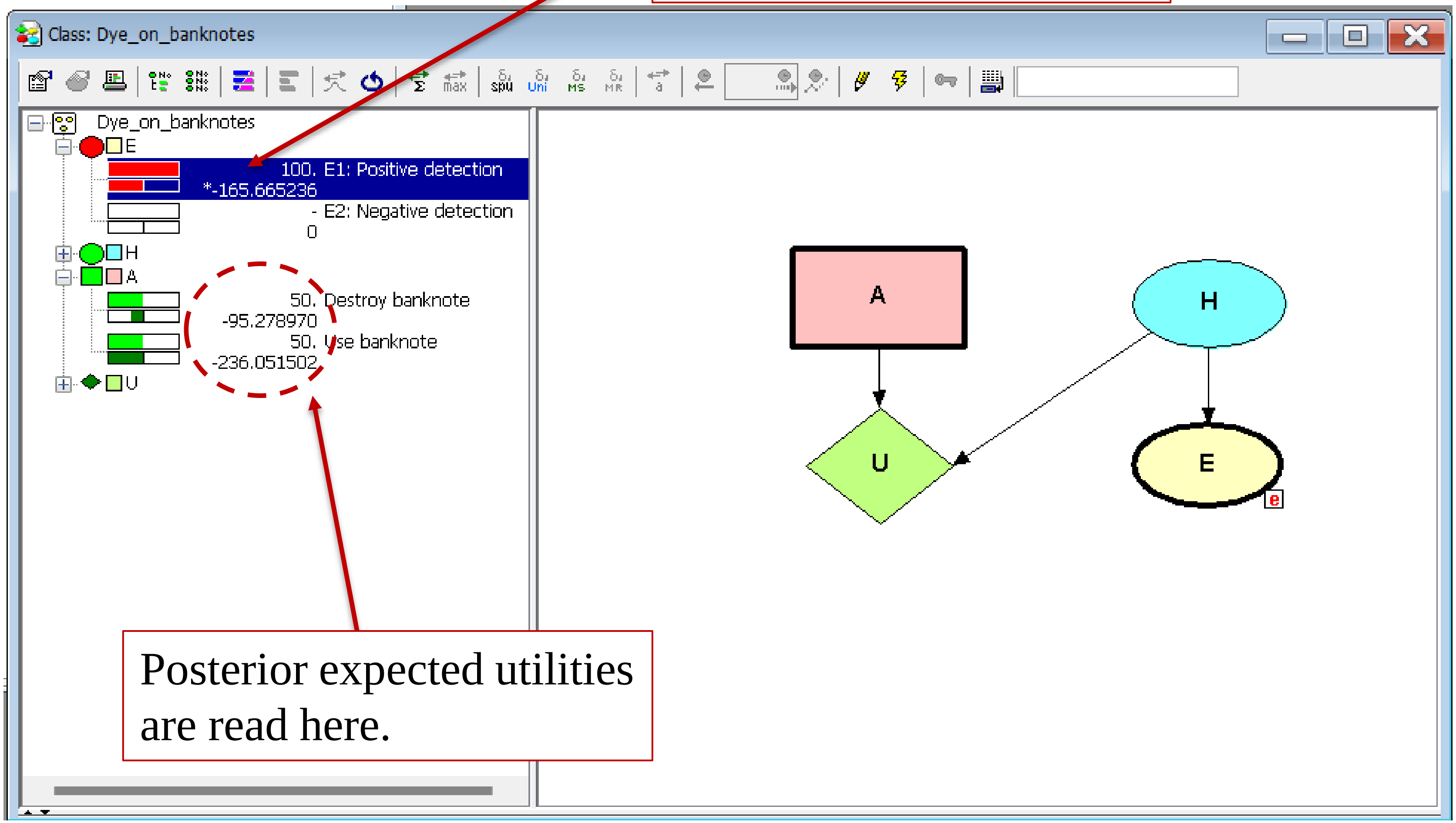




Class: Dye_on_banknotes
Dye_on_banknotes
E
H
A
50. Destroy banknote
-99.9
50. Use banknote
-5.
U
A
H
U
E
Prior expected utilities are read here.

Instantiating node $\boldsymbol{E}$ to state E_1:

Done my double-clicking on that state in the tree.



Posterior expected utilities are read here.

Example: Newcomb's problem

Recall Newcomb's problem from Meeting 9:

You are exposed to two "boxes".

In Box 1 you can see that there is an amount of $1000.
You cannot see what is in Box 2, but you are told that it is either nothing or $1 000 000.

You are offered to make one the following two choices:

- A1: Take <u>both</u> boxes
- A2: Take Box 2

<u>Before</u> you make your choice, a prediction expert will predict which choice you will make.
If her prediction is that your choice will be A2, $ 1 000 000 will be put in Box 2
If her prediction is that your choice will be A1, nothing will be put in Box 2.

The prediction expert's accuracy is 99%, i.e. she has been right in 99% of her predictions.

Decision matrix:

	States	
Actions	Prediction is A1	Prediction is A2
A1: Take both boxes	$ 1 000	$1 001 000
A2: Take Box 2	$ 0	$ 1 000 000

State probabilities depend on the action taken:

$$P(\text{Prediction is A1}|\text{Action is A1}) = 0.99$$

$$P(\text{Prediction is A2}|\text{Action is A1}) = 0.01$$

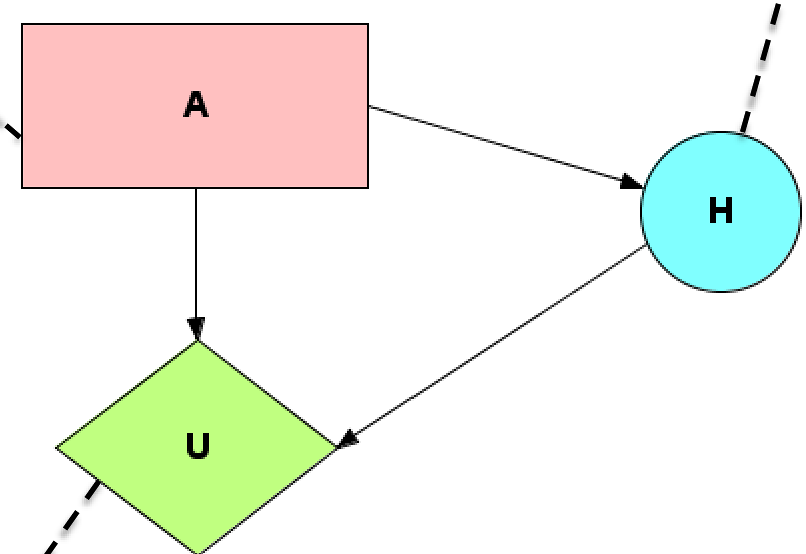
$$P(\text{Prediction is A1}|\text{Action is A2}) = 0.01$$

$$P(\text{Prediction is A2}|\text{Action is A2}) = 0.99$$

Influence diagram

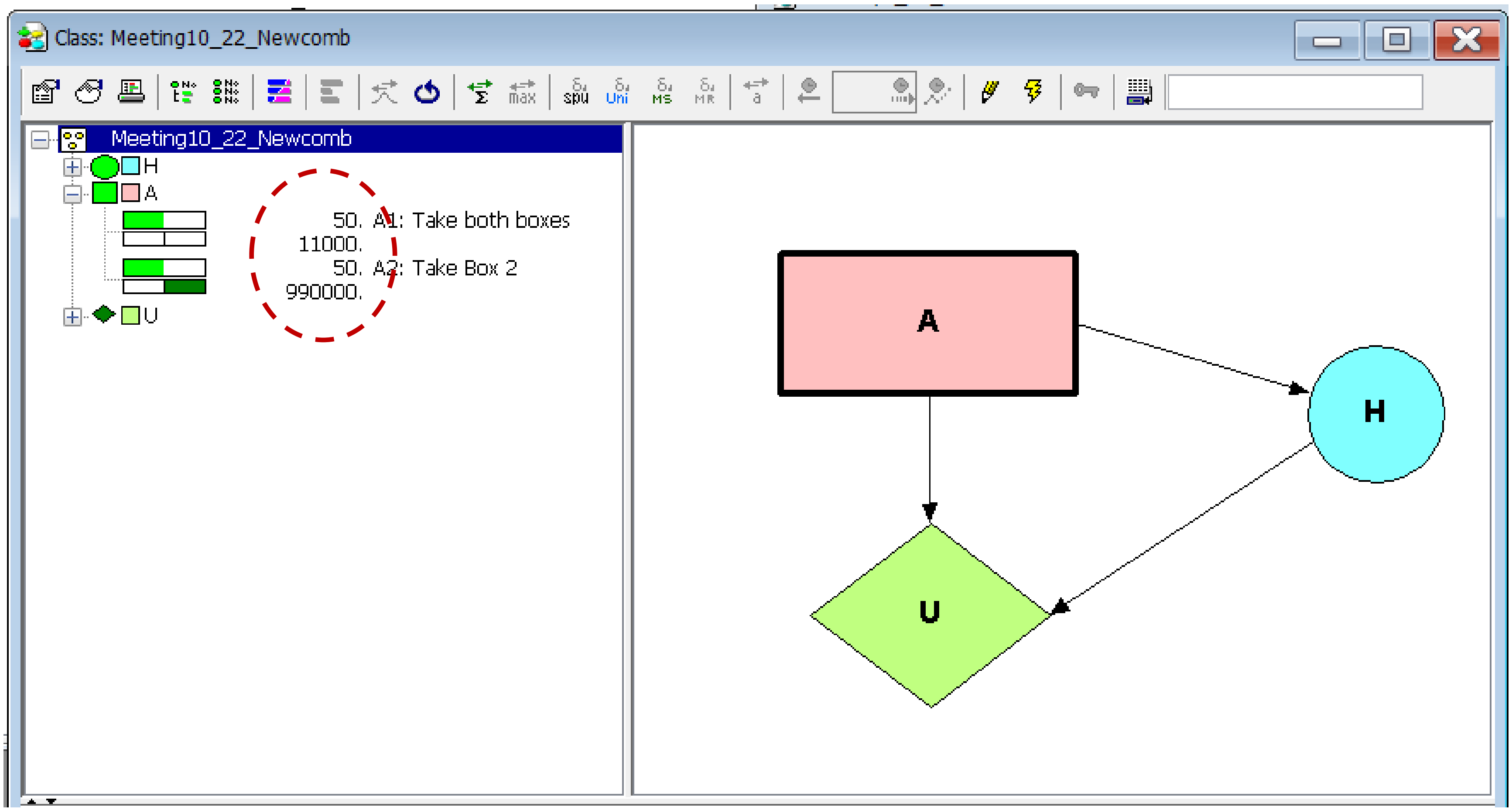
A	
A1	Take both boxes
A2	Take Box 2

	A:	A1	A2
H:	Predict A1	0.99	0.01
	Predict A2	0.01	0.99



A:	A1		A2	
H:	Predict A1	Predict A2	Predict A1	Predict A2
U:	1 000	1 001 000	0	1 000 000

Using HuginLite



$990\,000 > 11\,000 \Rightarrow$ Take Box 2!

Extending the problem

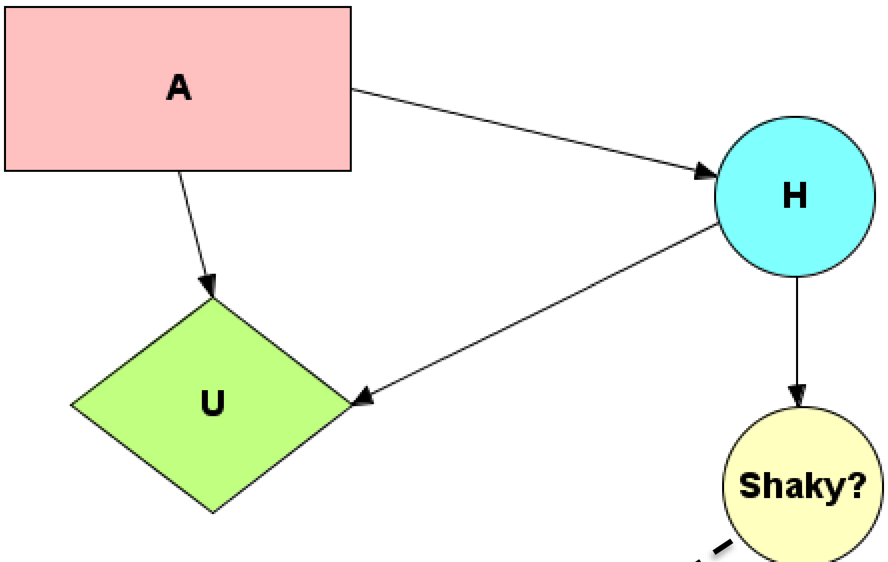
Assume that the person presenting these choice to you tends to be shaky in his hands when the predicting person has predicted A2 (i.e. when $ 1 000 000 has been put in Box 2.)

You were told that in 2 out of 5 cases when A2 is predicted, the presenter are shaky in his hands, while this happens in 1 out of 10 cases when A1 is predicted.

Watching the presenter, you <u>cannot</u> see that he is shaking.

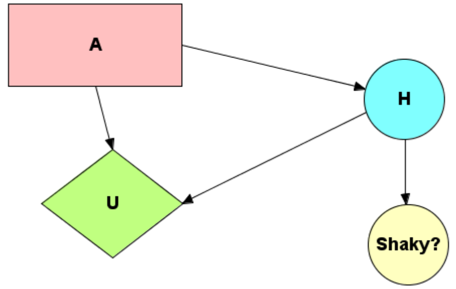
Would this affect your choice?

Try adding a new chance node to the network, that represents the “evidence” (data, observation)



	H:	Predict A1	Predict A2
Shaky?	Not shaky	0.9	0.6
	Shaky	0.1	0.4

	H:	Predict A1	Predict A2
Shaky?	Not shaky	0.9	0.6
	Shaky	0.1	0.4



Upon running the model, state “Not shaky” should be instantiated and this will update the probabilities in node **H**:

	A:	A1	A2
H:	Predict A1	0.99	0.01
	Predict A2	0.01	0.99

$$P(\text{Predict A1}|\text{Not shaky}, \text{A}j) =$$

$$= \frac{P(\text{Not shaky}|\text{Predict A1}, \text{A}j) \cdot P(\text{Predict A1}|\text{A}j)}{[P(\text{Not shaky}|\text{Predict A1}, \text{A}j) \cdot P(\text{Predict A1}|\text{A}j) + P(\text{Not shaky}|\text{Predict A2}, \text{A}j) \cdot P(\text{Predict A2}|\text{A}j)]}, \qquad j = 1{,}2$$

Analogously for $P(\text{Predict A2}|\text{Not shaky}, \text{A}j)$

	A:	A1	A2
H:	Predict A1	$\frac{0.9 \cdot 0.99}{0.9 \cdot 0.99 + 0.6 \cdot 0.01} \approx 0.9933$	$\frac{0.9 \cdot 0.01}{0.9 \cdot 0.01 + 0.6 \cdot 0.99} \approx 0.0149$
	Predict A2	$\frac{0.6 \cdot 0.01}{0.9 \cdot 0.99 + 0.6 \cdot 0.01} \approx 0.0067$	$\frac{0.6 \cdot 0.99}{0.9 \cdot 0.01 + 0.6 \cdot 0.99} \approx 0.9851$

With the updated probability table of node **H**,

	A:	A1	A2
H:	Predict A1	0.9933	0.0149
	Predict A2	0.0067	0.9851

we calculate posterior expected utilities:

A:	A1		A2	
H:	Predict A1	Predict A2	Predict A1	Predict A2
U:	1 000	1 001 000	0	1 000 000

$$E(U(\text{A1})|\text{Not shaky}) = 1000 \cdot 0.9933 + 1001000 \cdot 0.0067 = 7700$$

$$E(U(\text{A2})|\text{Not shaky}) = 0 \cdot 0.0149 + 1000000 \cdot 0.9851 = 985100$$

985 100 > 7 000 ⇒ Still take Box 2!

But, can the state probabilities be updated this way?