Decision Networks (Influence Diagrams)

CS 486/686: Introduction to Artificial Intelligence
Fall 2013

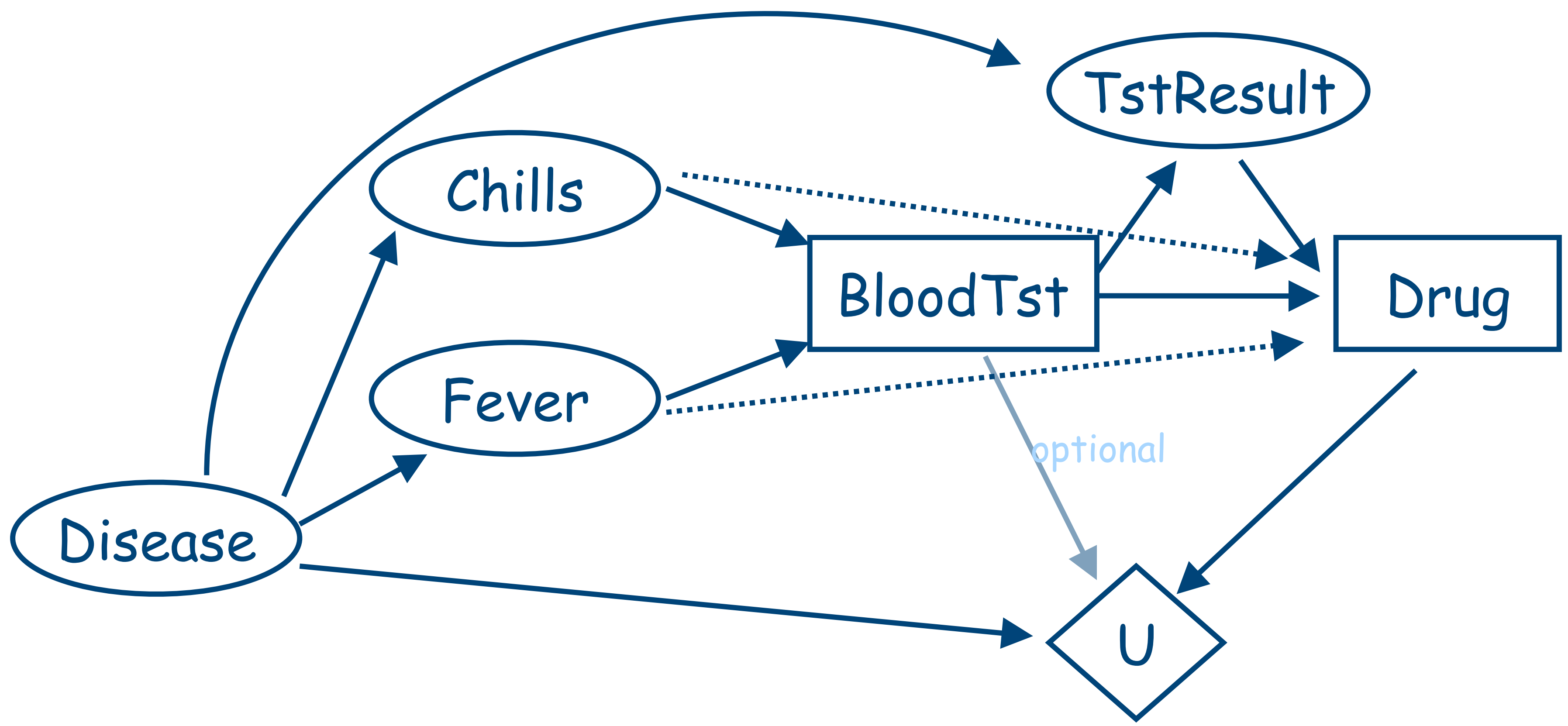
Outline

- Decision Networks
- Computing Policies
- Value of Information

Introduction

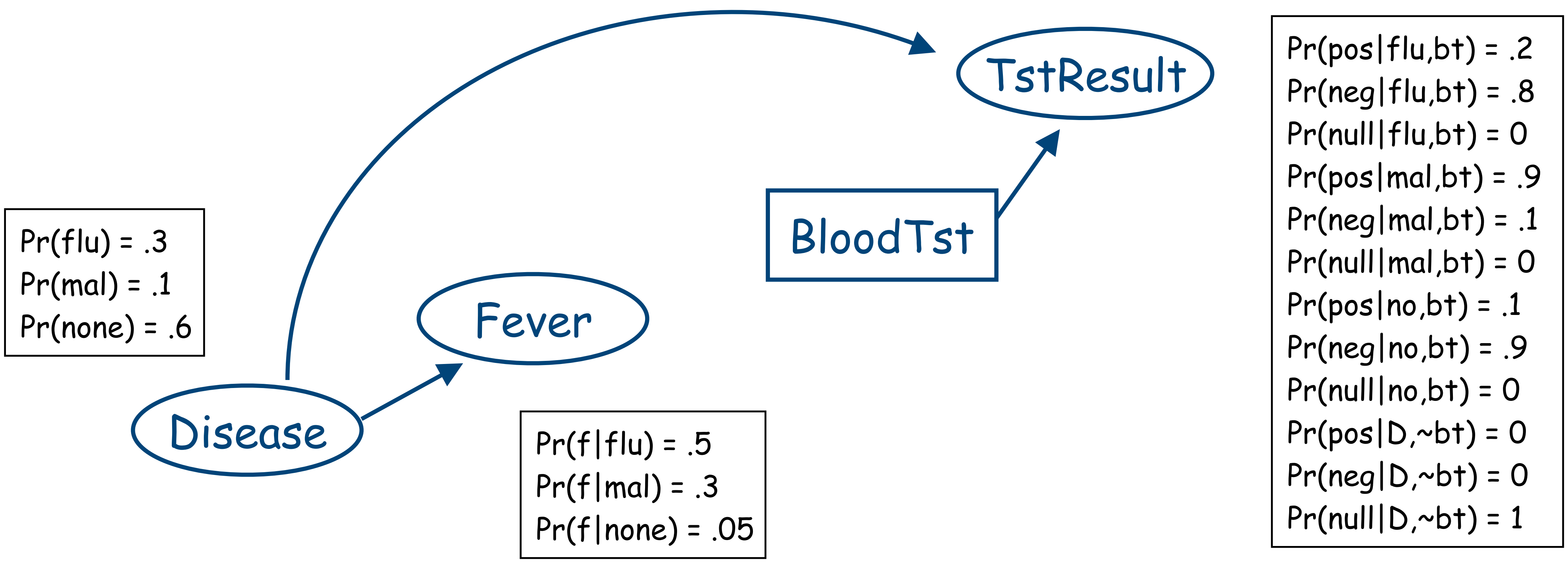
- Decision networks (aka influence diagrams) provide a representation for sequential decision making
- Basic idea
  - Random variables like in Bayes Nets
  - Decision variables that you “control”
  - Utility variables which state how good certain states are

Example Decision Network



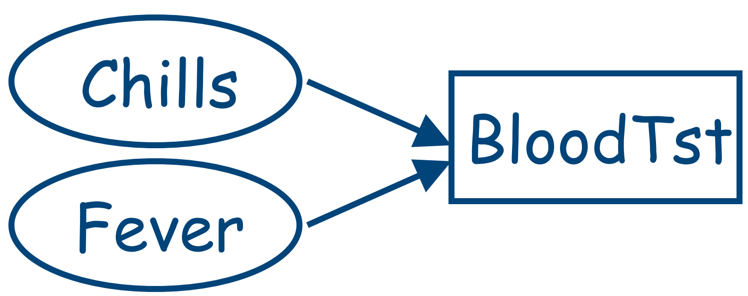
Chance Nodes

- Random variables (denoted by circles)
- Like as in a BN, probabilistic dependence on parents



Decision Nodes

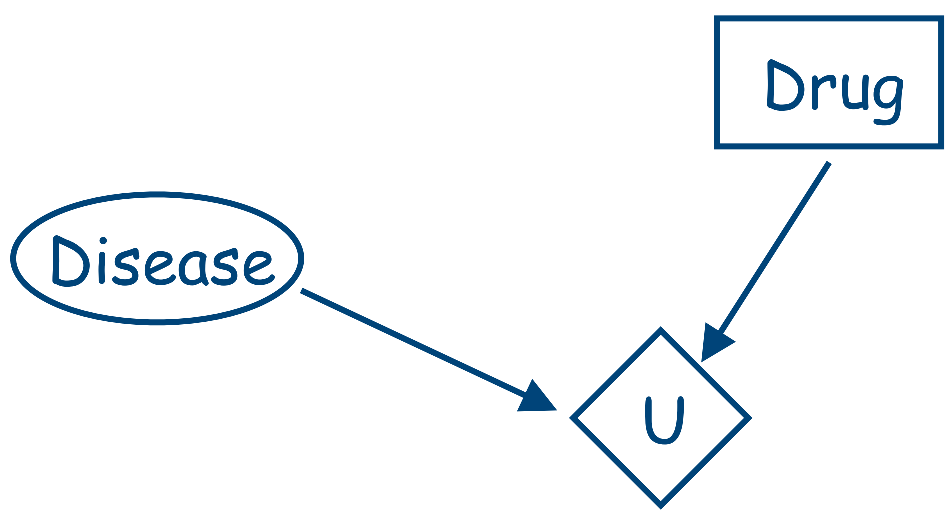
- Variables the decision maker sets (denoted by squares)
- Parents reflect information available at time of decision



BT $\in$ {bt, ~bt}

Value Nodes

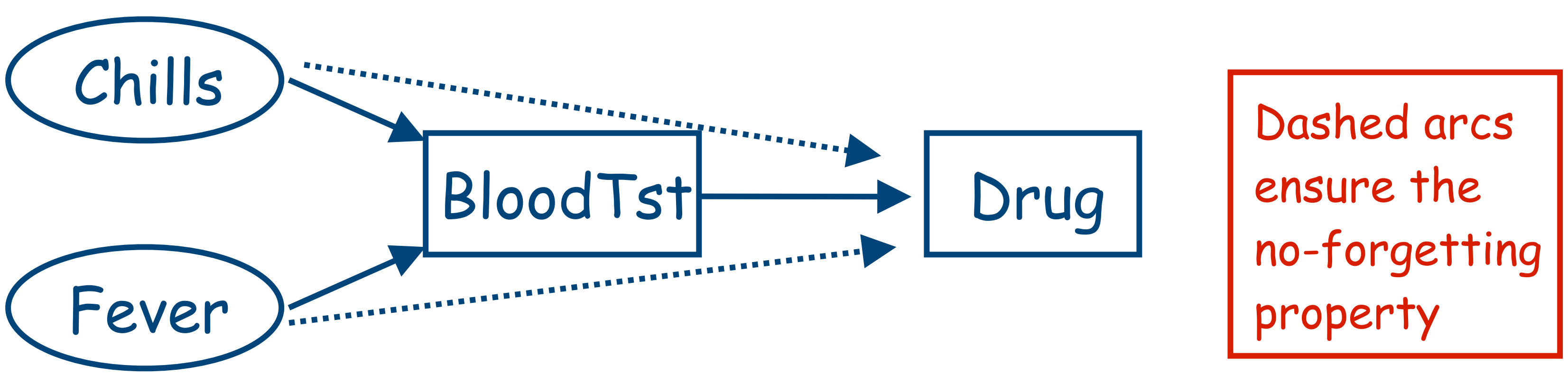
- Specifies the utility of a state (denoted by a diamond)
- Utility depends only on state of parents
- Generally, only one value node in a network



U(fludrug, flu) = 20
U(fludrug, mal) = -300
U(fludrug, none) = -5
U(maldrug, flu) = -30
U(maldrug, mal) = 10
U(maldrug, none) = -20
U(no drug, flu) = -10
U(no drug, mal) = -285
U(no drug, none) = 30

Assumptions

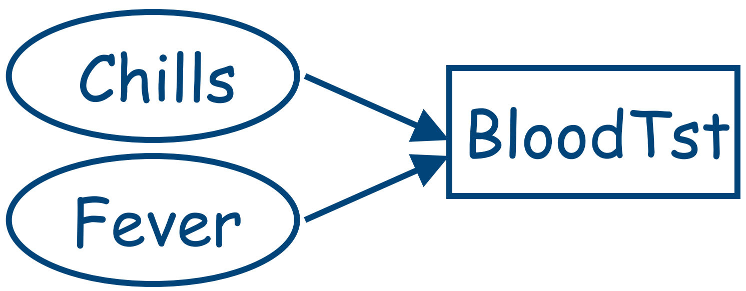
- Decision nodes are totally ordered
  - Given decision variables $D_1, \ldots, D_n$, decisions are made in sequence
- No forgetting property
  - Any information available for decision D_i remains available for decision D_j where $j>i$
  - All parents of D_i are also parents for D_j



Policies

- Let $Par(D_i)$ be the parents of decision node D_i
  - $Dom(Par(D_i))$ is the set of assignments to $Par(D_i)$
- A policy δ is a set of mappings δ_i, one for each decision node D_i
  - $\delta_i(D_i)$ associates a decision for each parent assignment
  - $\delta_i : Dom(Par(D_i)) \rightarrow Dom(D_i)$

$\delta_{BT}(c,f)=bt$
$\delta_{BT}(c,\sim f)=\sim bt$
$\delta_{BT}(\sim c,f)=bt$
$\delta_{BT}(\sim c,\sim f)=\sim bt$



Value of a Policy

- The value of a policy δ is the expected utility given that decision nodes are executed according to δ
- Given assignment **x** to random variables **X**, let δ(**x**) be the assignment to decision variables dictated by δ
  - Value of δ

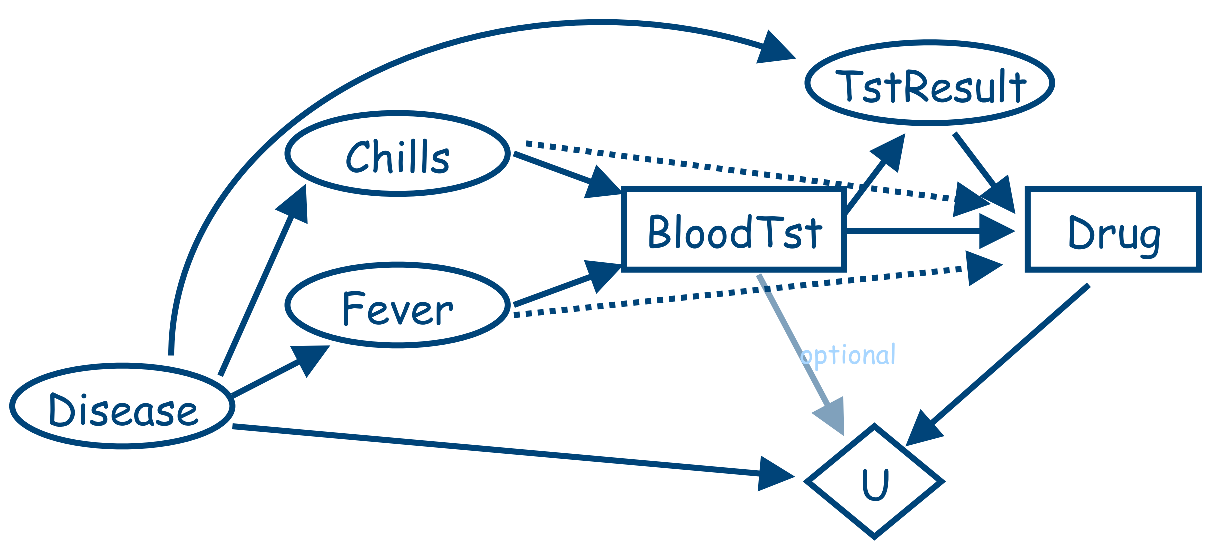
$$EU(\delta)=\sum_{\mathbf{x}} P(\mathbf{x},\delta(\mathbf{x}))U(\mathbf{x},\delta(\mathbf{x}))$$

Optimal Policy

- An optimal policy δ^* is such that $EU(\delta^*) \geq EU(\delta)$ for all δ
- We can use dynamic programming to avoid enumerating all possible policies
- We can also use the BN structure and Variable Elimination to aid the computation

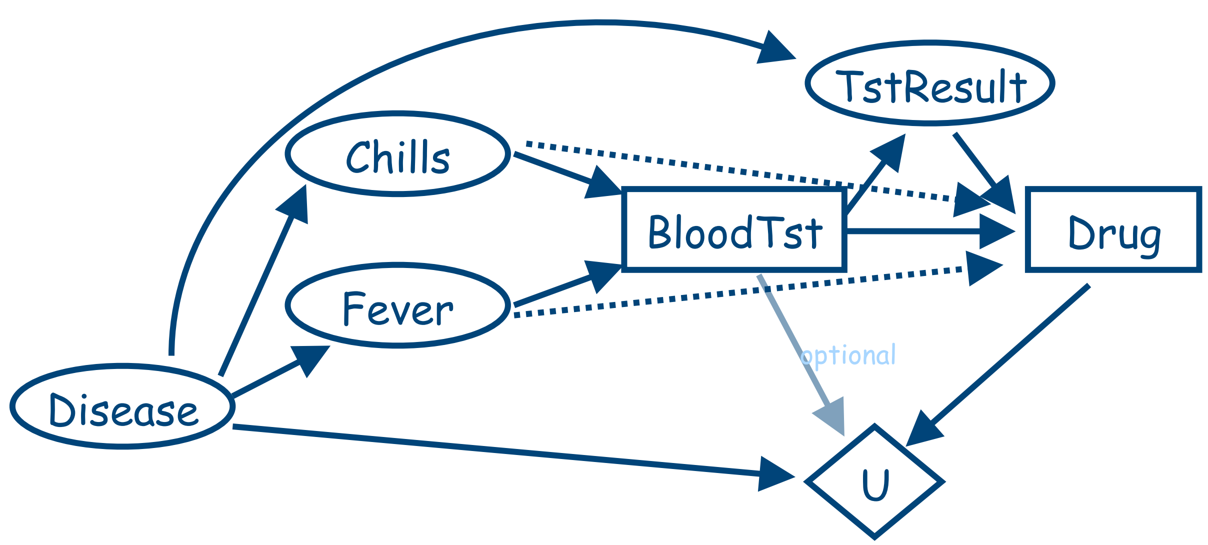
Computing the Optimal Policy

- Work backwards as follows
  - Compute optimal policy for Drug
    - For each asst to parents (C,F,BT,TR) and for each decision value (D = md,fd,none), **compute the expected value** of choosing that value of D
    - Set policy choice for each value of parents to be the value of D that has max value



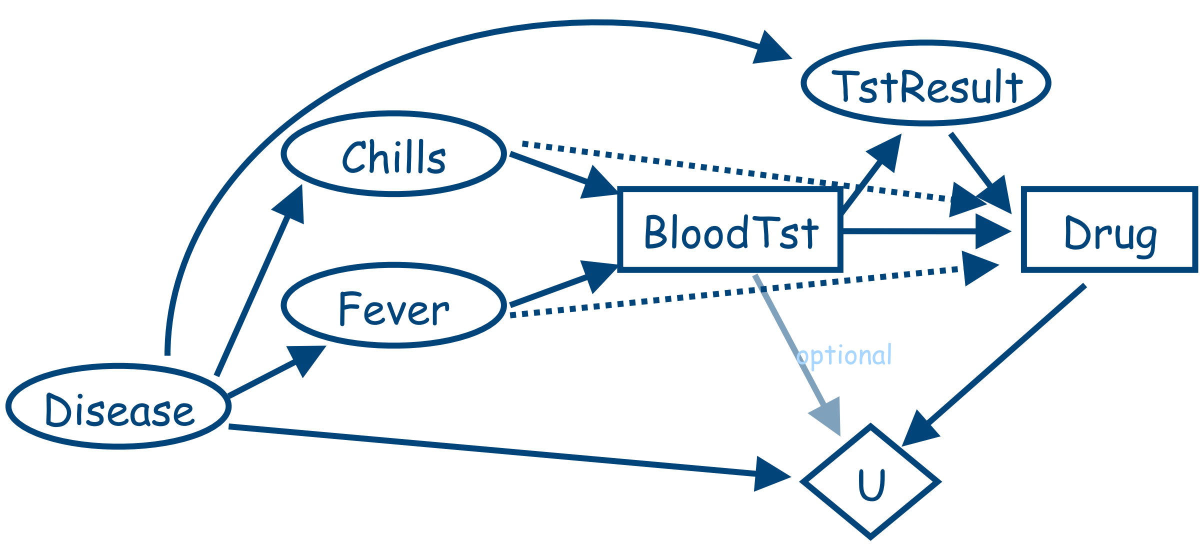
Computing the Optimal Policy

- Next compute policy for BT, given policy δ_D(C,F,BT,TR) just computed
  - Since δ_D is fixed, we treat D as a random variable with deterministic probabilities
  - Solve for BT just like you did for D



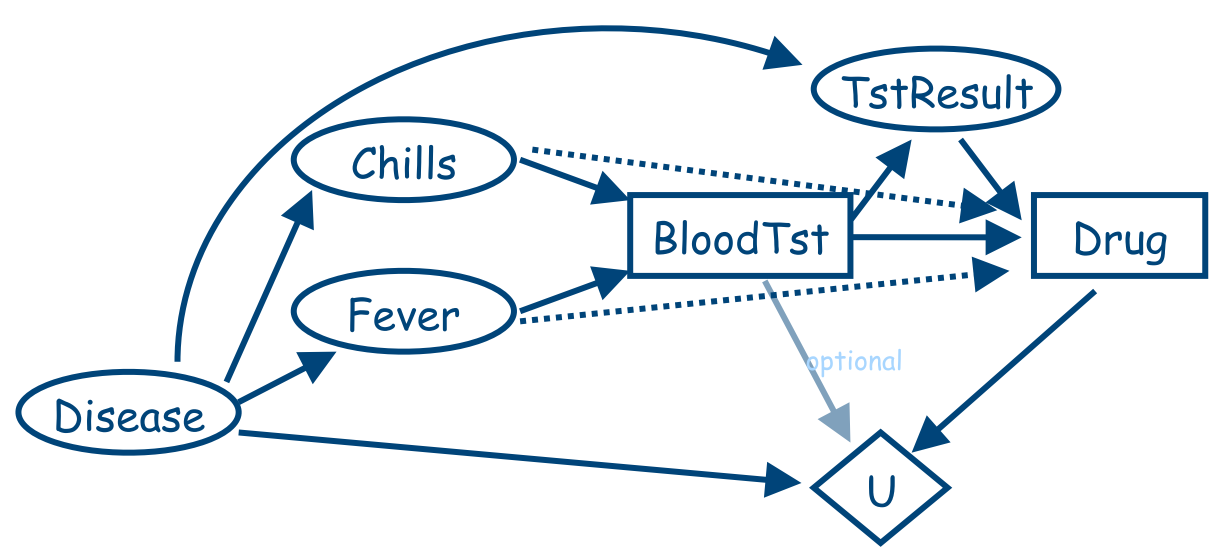
Computing the Optimal Policy

- How do we compute these expected values?
  - Suppose we have asst *<c,f,bt,pos>* to parents of *Drug*
  - We want to compute EU of deciding to set *Drug = md*
  - We can run variable elimination!



Computing the Optimal Policy

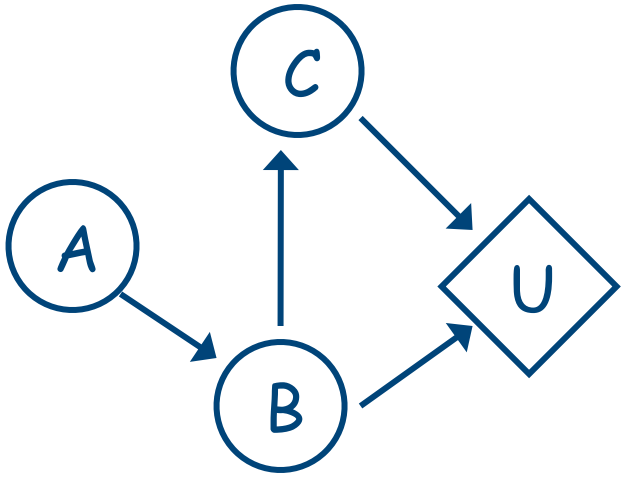
- Treat C, F, BT, Tr, Dr as evidence
  - This reduces the factors
  - Eliminate remaining variables (Dis)
  - Left with factor $U() = \Sigma_{Dis}$ P(Dis l c,f,bt,pos,md)U(Dis,md,bt)
- We now know EU of doing Dr=md when c,f,bt,pos



Computing Expected Utilities

- Computing expected utilities with BNs is straightforward
- Utility nodes are just factors that can be dealt with using variable elimination

$$EU = \Sigma_{A,B,C}\ P(A,B,C)\ U(B,C)$$
$$= \Sigma_{A,B,C}\ P(C|B)\ P(B|A)\ P(A)\ U(B,C)$$



Optimizing Policies: Key Points

- If decision node D has no decisions that follow it, we can find its policy by instantiating its parents and computing the expected utility for each decision given parents
  - No-forgetting means that all other decision are instantiated
  - Easy to compute the expected utility using VE
  - Number of computations is large
    - We run expected utility calculations for each parent instantiation and each decision instantiation
  - Policy: Max decision for each parent instantiation

Optimizing Policies: Key points

- When node D is optimized, can be treated as a random variable
- If we optimize from the last decision to the first, at each point we can optimize a single decision by simple VE
  - Why? Its successor decisions are simply random variables in the BN

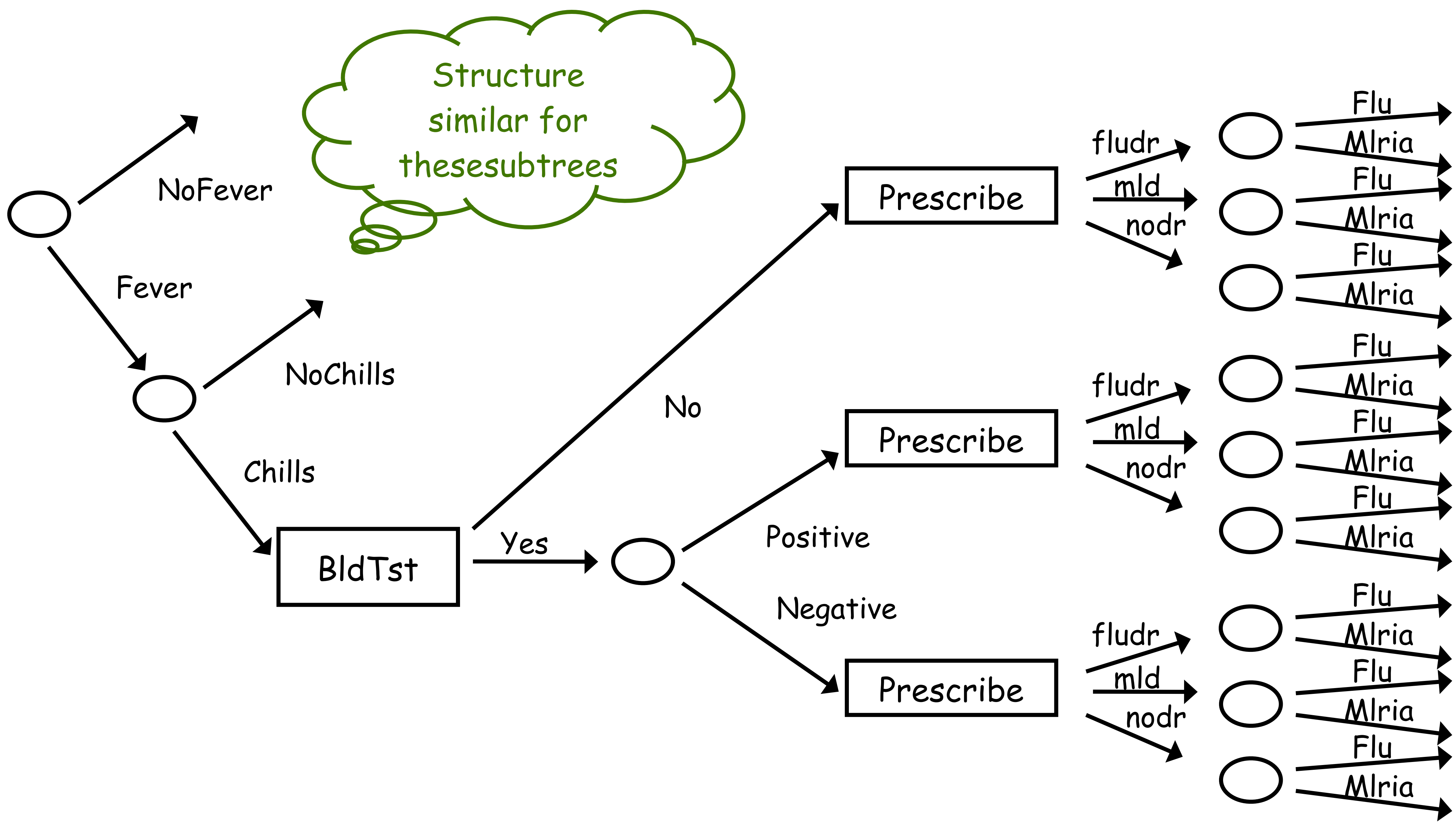
Notes

- Commonly used by decision analysts to help structure decision problems
- Much work put into computationally effective techniques to solve them
  - Common trick: replace decision nodes with random variables at the outset and solve a plain BN
- Complexity is much greater than BN inference

Decision Trees and Decision Networks

- It is possible to build a decision tree from a decision network
  - Order decisions as in the network
  - Ensure that observed chance nodes appear before decisions that use them
  - Label leaves with utilities dictated from utility nodes
  - Assign probabilities to outcomes using conditional probabilities of outcomes given observed variables and decisions on the branch so far

Decision Tree for Medical Network

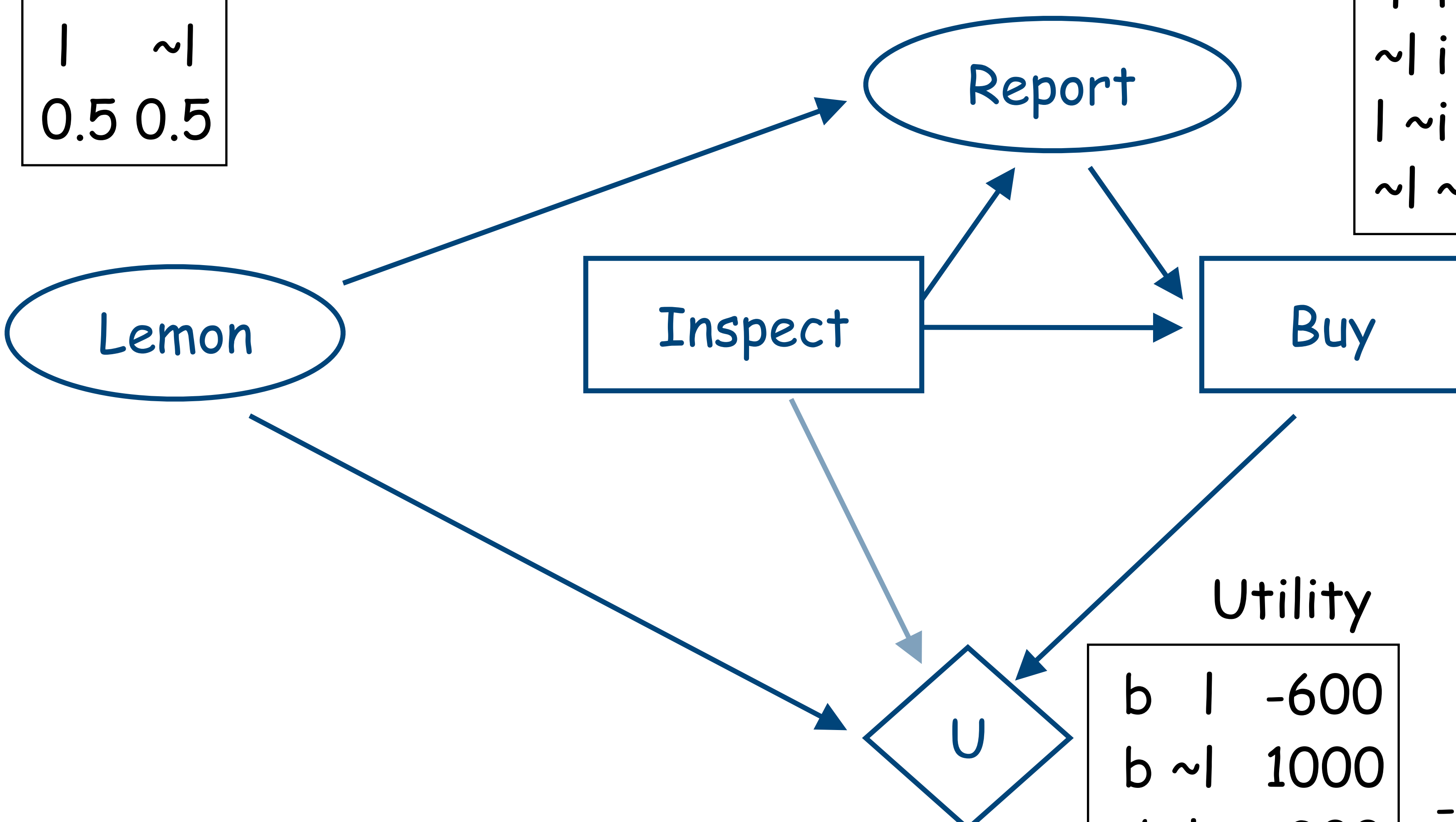


Example: Decision Network

- You want to buy a used car, but there is some chance it is a "lemon" (i.e. it breaks down often). Before deciding to buy it, you can take it to a mechanic for an inspection. S/he will give you a report, labelling the car as either "good" or "bad". A good report is positively correlated with the car not being a lemon while a bad report is positively correlated with the car being a lemon
- The report costs $50. You could risk it and buy the car with no report.
- Owning a good car is better than no car, which is better than owning a lemon.

Example

l	~l
0.5	0.5



Rep: good,bad,none

	g	b	n
l i	0.2	0.8	0
~l i	0.9	0.1	0
l ~i	0	0	1
~l ~i	0	0	1

Utility

b l	-600
b ~l	1000
~b l	-300
~b~l	-300

-50 if inspect

Value of Information

- Claim: Optimal policy is “Inspect car, buy if the report is good” (EU=205)
  - Note that the EU of inspecting the car and buying if you get a good report is 255 minus the cost of the inspection (50)
- At what point would you no longer be interested in doing the inspection?
  - Find V(I) such that $255 - V(I) \leq EU(\sim i) = 200$
- The **expected value of information** associated with the inspection is $55
  - You should be willing to pay up to $55 for the inspection

Value of Information

- Information has value
  - To the extent it is likely to cause a change of plan
  - To the extent that the new plan will be significantly better than the old plan
- The value of information is non-negative
  - This is true for any decision-theoretic agent

Summary

- Definition of a Decision Network
- Definition of an Optimal Policy
- Computing Optimal Policies
- Relationship between DN and DT
- Value of Information