Decision Theory: Sequential Decisions

CPSC 322 Lecture 33

Recap "Single" Action

Set of primitive decisions that can be treated as a single macro decision to be made *before acting*

- eg. A robot deciding which way to go and whether to wear pads can treat it as a single decision with options made from the options of the constituent decisions (such as "long way with pads on")
- The decision is not dependent on information from a random variable

Learning Goals for today's class

You can:

- Represent **sequential decision problems** as decision networks. And explain the **non forgetting property**
- Verify whether a **possible world satisfies a policy** and define the **expected value of a policy**
- Compute the **number of policies** for a decision problem
- **Compute the optimal policy** by Variable Elimination

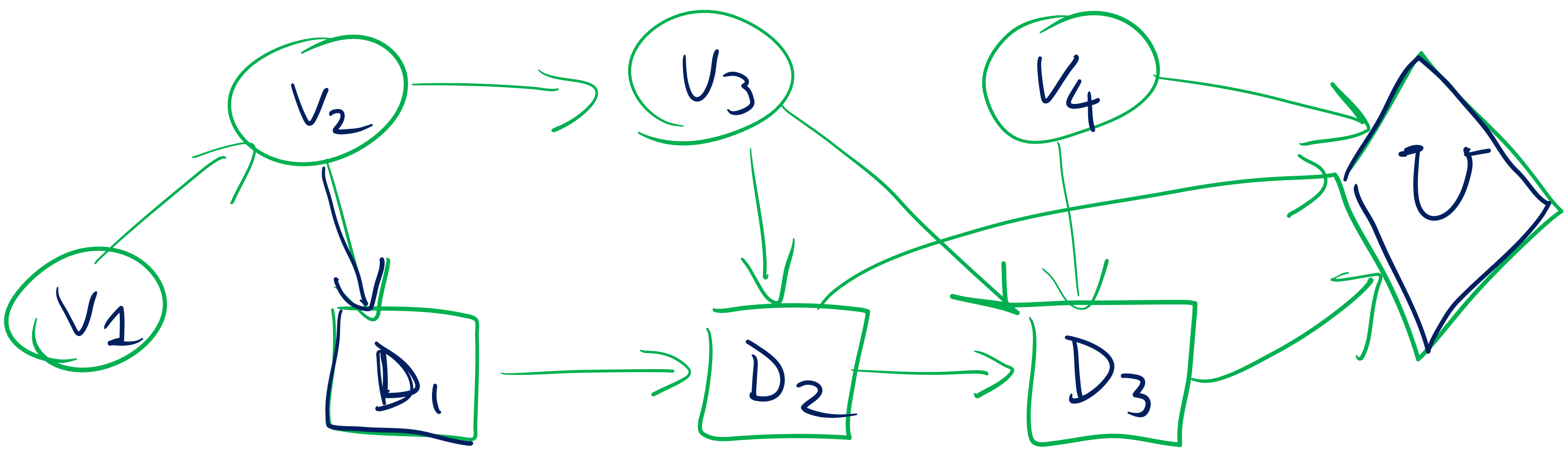
Lecture Overview

- Sequential Decisions
  - Representation
  - Policies
- Finding Optimal Policies

Sequential decision problems

- A sequential decision problem consists of a sequence of decision variables $D_1, \ldots\ldots, D_n$.
- Each D_i has an information set of variables pD_i, whose value will be known at the time decision D_i is made.

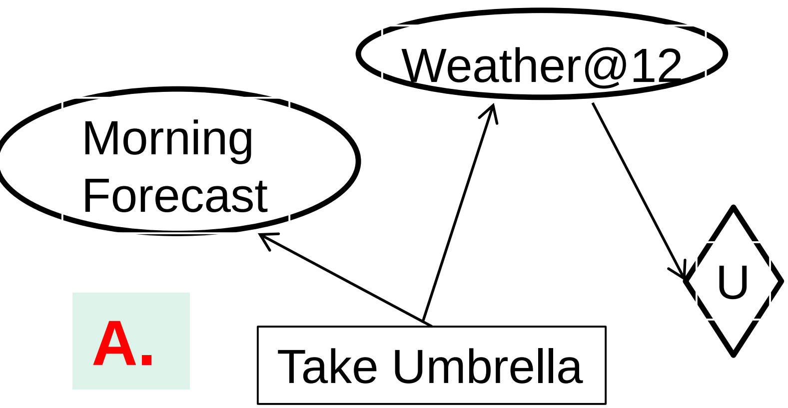
$pD_3 = \{D_2 \; V_3 \; V_4\}$



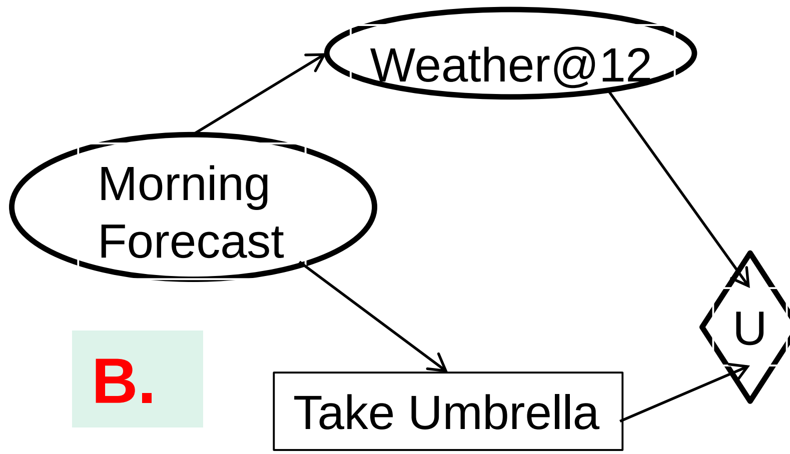
Sequential decisions : Simplest possible

- Only one decision! (but different from one-off decisions)
- Early in the morning. I listen to the **weather forecast**, shall I take my **umbrella** today? (I'll have to go for a long walk **at noon**)
- What is a reasonable decision network ?

A.



B.



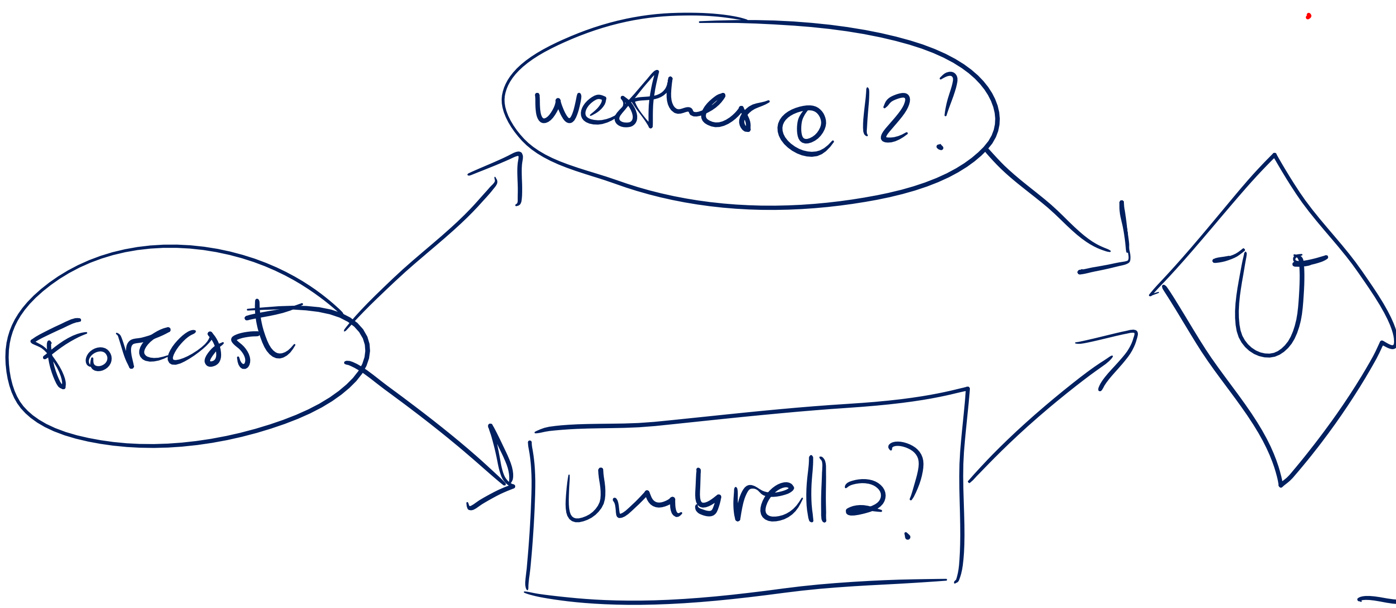
C . None of these

D. Both of these

E. The answer has to be 42 eventually…

Sequential decisions : Simplest possible

- Only one decision! (but different from one-off decisions)
- Early in the morning. Shall I take my **umbrella** today? (I'll have to go for a long walk at noon)
- Relevant Random Variables?



Policies for Sequential Decision Problem: Intro

- A **policy** specifies what an agent should do under each circumstance (for each decision, consider the parents of the decision node)

In the *Umbrella* “degenerate” case:

D_1 ? T F

One possible Policy

pD_1

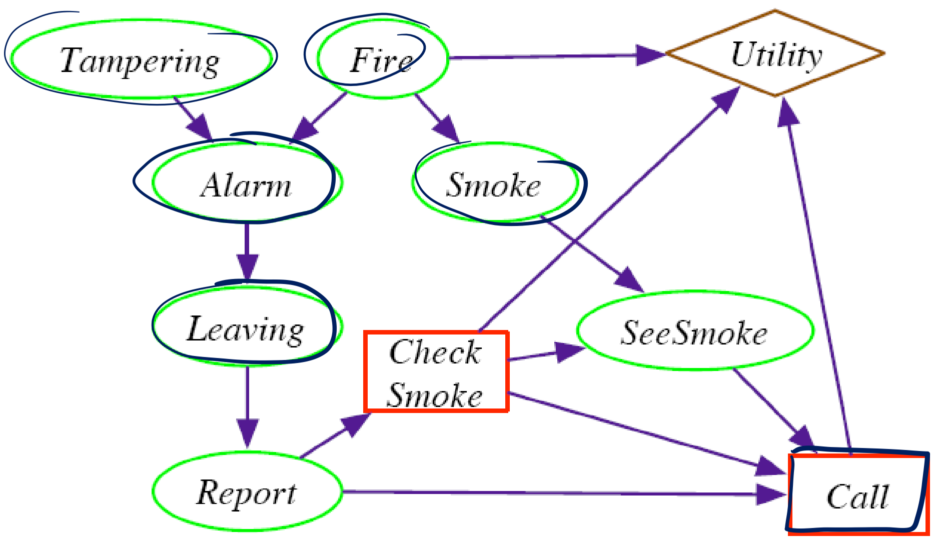
Rainy	R	T	F	T ...
Cloudy	C	T	F	T ...
Sunny	S	F	F	T ...

3 policies

How many policies? 2^3

Sequential decision problems: "complete" Example

- A sequential decision problem consists of a sequence of decision variables $D_1, \ldots\ldots, D_n$.
- Each D_i has an information set of variables pD_i, whose value will be known at the time decision D_i is made.



$pCS = \{R\}$
$pC = \{R, CS, SS\}$

No-forgetting decision network:

- decisions are totally ordered
- if a decision D_b comes before D_a ,then
  - D_b is a parent of D_a
  - any parent of D_b is a parent of D_a

Policies for Sequential Decision Problems

- A **policy** is a sequence of $\delta_1, \ldots, \delta_n$ **decision functions**

$$\delta_i : \mathrm{dom}(pD_i) \rightarrow \underline{\mathrm{dom}(D_i)}$$

- This policy means that when the agent has observed $O \in \mathrm{dom}(pD_i)$, it will do $\delta_i(O)$

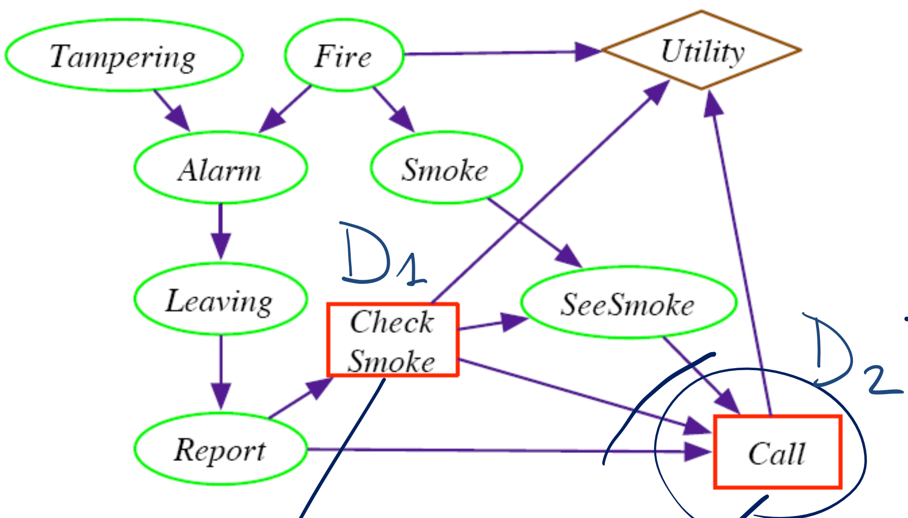
Example: δ_1

Report	Check Smoke	
T	T T F	F
F	T F T	F

δ_2

Report	CheckSmoke	SeeSmoke	Call
true	true	true	true
true	true	false	false
true	false	true	true
true	false	false	false
false	true	true	true
false	true	false	false
false	false	true	false
false	false	false	false

8



How many policies?

$2^2 * 2^8$

Lecture Overview

- Recap
- Sequential Decisions
- Finding Optimal Policies

When does a possible world satisfy a policy?

- A possible world specifies a value for each random variable and each decision variable.
- **Possible world** w **satisfies policy** δ, written $w \models \delta$ if the value of each decision variable is the value selected by its decision function in the policy (when applied in w).

This world does not satisfy the policy – why?

VARs	
Fire	true
Tampering	false
Alarm	true
Leaving	true
Report	false
Smoke	true
SeeSmoke	true
CheckSmoke	true
Call	true

Decision function for CheckSmoke

Report	**CheckSmoke**
true	true
false	false

Decision function for Call

Report	**CheckSmoke**	**SeeSmoke**	**Call**
true	true	true	true
true	true	false	false
true	false	true	true
true	false	false	false
false	true	true	true
false	true	false	false
false	false	true	false
false	false	false	false

When does a possible world satisfy a policy?

- Possible world *w* satisfies policy δ , written $w \models \delta$ if the value of each decision variable is the value selected by its decision function in the policy (when applied in *w*).

w_1

VARs	
Fire	true
Tampering	false
Alarm	true
Leaving	true
Report	true
Smoke	true
SeeSmoke	true
CheckSmoke	true
Call	true

Decision function for CheckSmoke δ

Report	**CheckSmoke**
true	true
false	false

Decision function for Call

Report	**CheckSmoke**	**SeeSmoke**	**Call**
true	true	true	true
true	true	false	false
true	false	true	true
true	false	false	false
false	true	true	true
false	true	false	false
false	false	true	false
false	false	false	false

A. $w_1 \models \delta$

B. $w_1 \not\models \delta$

C . Cannot tell

iclicker.

Expected Value of a Policy

- Each possible world w has a probability $P(w)$ and a utility $U(w)$
- The expected utility of policy δ is

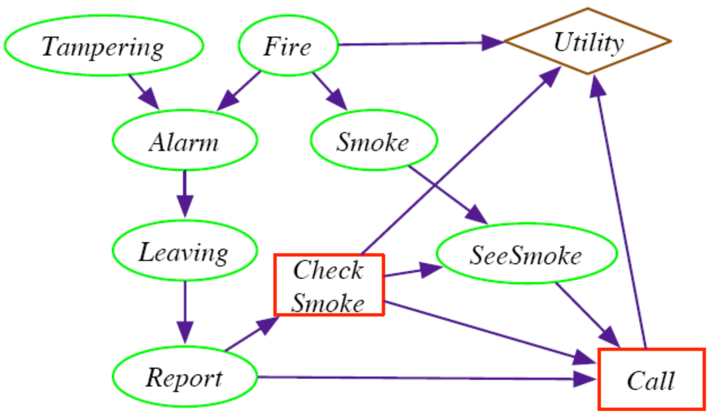
$$\sum_{w \models \delta} P(w) * U(w)$$

- The optimal policy is one with the max expected utility.

Lecture Overview

- Recap
- Sequential Decisions
- Finding Optimal Policies (Efficiently)

Complexity of finding the optimal policy: how many policies?



- How many assignments to parents?

  CS 2 C 2^3

- How many decision functions? (binary decisions)

  2^2 2^{2^3}

- How many policies? product $2^2 * 2^{2^3}$

- If a decision *D* has *k* binary parents, how many assignments of values to the parents are there? 2^K

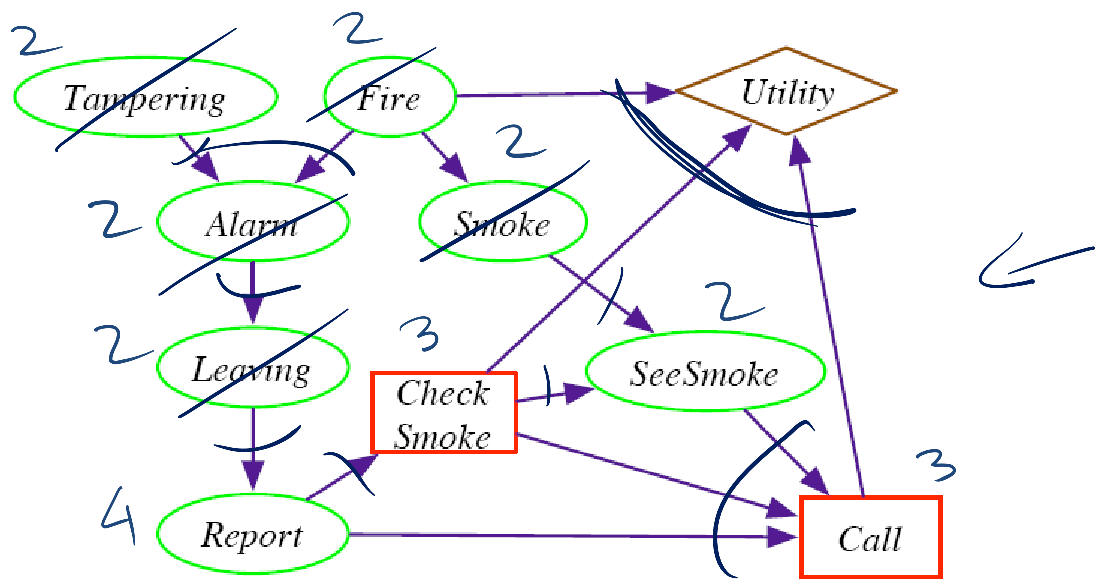
- If there are *b* possible actions (possible values for D), how many different decision functions are there? $\boxed{b^{2^K}}$

- If there are *d* decisions, each with *k* binary parents and *b* possible actions, how many policies are there? $\left(b^{2^K}\right)^d$

Finding the optimal policy more efficiently: VE

1. Create a factor for each conditional probability table and a factor for the utility.
2. **Sum out** random variables that are not parents of a decision node.
3. **Eliminate** (by maximization) the decision variables
4. **Sum out** the remaining random variables.
5. **Multiply the factors**: this is the expected utility of the optimal policy.

repeat as needed



Eliminate the decision Variables: step3 details

- Select a variable *D* that corresponds to the latest decision to be made
  - this variable will appear in only one factor with its parents
- Eliminate *D* by **maximizing**. This returns:
  - A new factor to use in VE, $\max_D f$
  - The optimal decision function for *D*, $\arg\max_D f$
- Repeat till there are no more decision nodes.

Example: Eliminate CheckSmoke

Report	CheckSmoke	Value
true	true	-5.0
true	false	-5.6
false	true	-23.7
false	false	-17.5

New factor

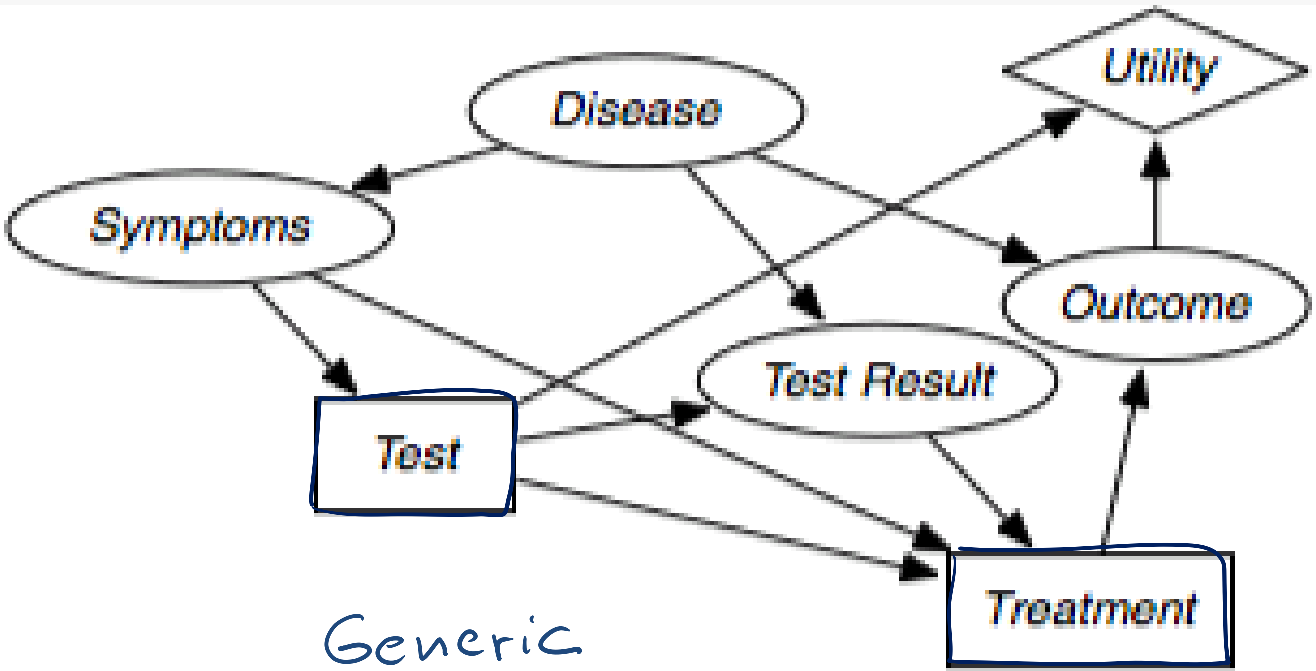
Report	Value
true	-5.0
false	-17.5

Decision Function

Report	CheckSmoke
true	true
false	false

VE elimination reduces complexity of finding the optimal policy

- We have seen that, if a decision *D* has *k* binary parents, there are *b* possible actions, and there are d decisions,
- Then there are: $(b^{2^k})^d$ *policies*
- Doing variable elimination lets us find the optimal policy after considering only $d * b^{2^k}$ policies (we eliminate one decision at a time)
  - VE **is much more efficient** than searching through policy space.
  - However, this complexity is **still doubly-exponential** - we'll only be able to handle relatively small problems
    - ✓ Give up non-forgetting assumption
    - ✓ Use approximate inference algorithms (CPSC 422)

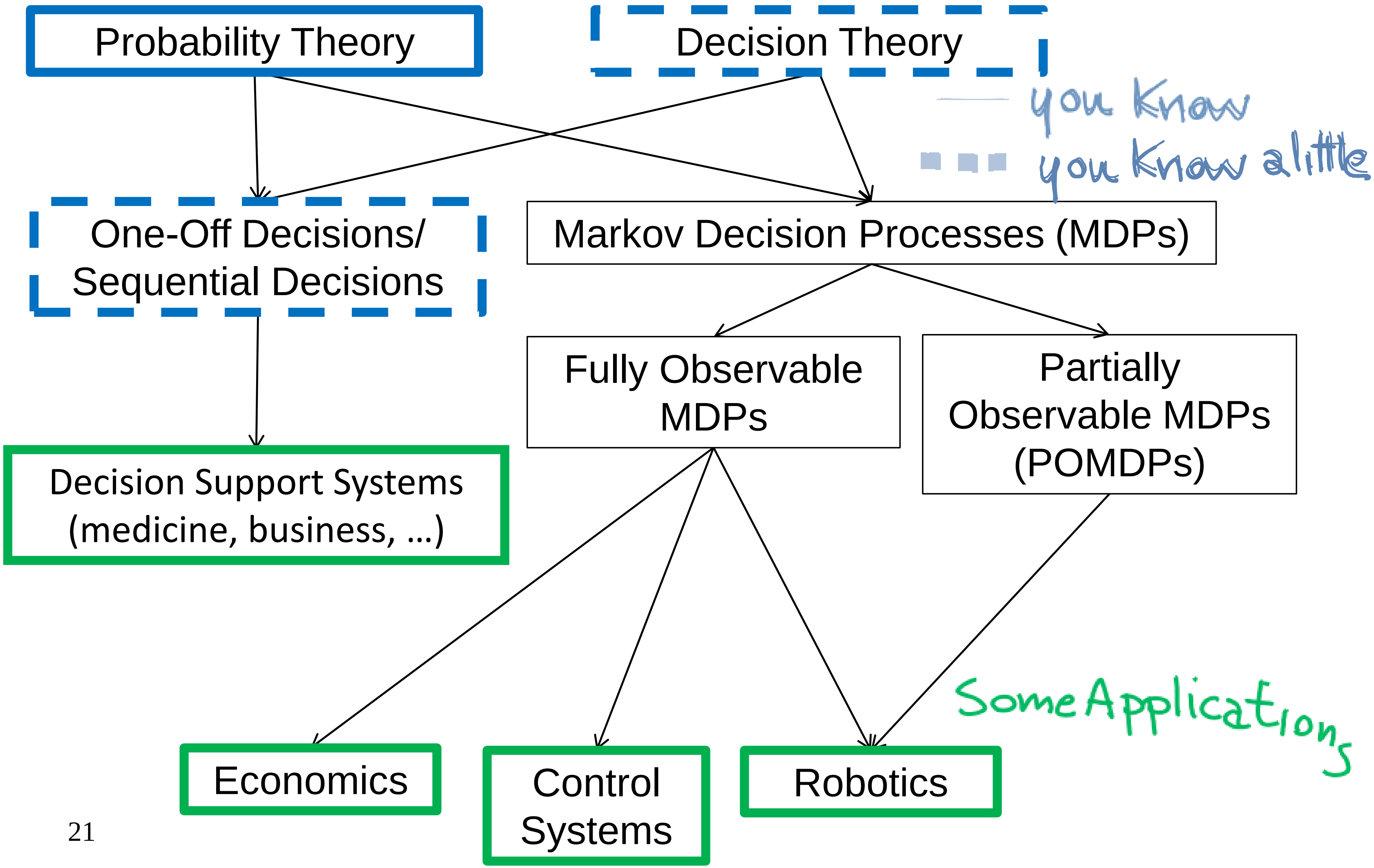


Generic

Figure 9.8: Decision network for diagnosis

to select what test to apply
and then what treatment to prescribe

Big Picture: Planning under Uncertainty



CPSC 322 Big Picture

		Environment	
Problem		**Deterministic**	**Stochastic**
Static	Constraint Satisfaction	*Variables + Constraints* Search Arc Consistency Local Search	
	Query	*Logics* Search	*Bayesian (Belief) Networks* Variable Elimination
Sequential	Planning	*STRIPS* Search	*Decision Networks* Variable Elimination

Representation
Reasoning Technique

422 big picture

StarAI (statistical relational AI)

Hybrid: Det +Sto

Prob CFG

Prob Relational Models

Markov Logics

Deterministic | **Stochastic**

Query

Logics

First Order Logics

Ontologies

- Full Resolution
- SAT

Belief Nets

Approx. : Gibbs

Markov Chains and HMMs

Forward, Viterbi….

Approx. : Particle Filtering

Undirected Graphical Models

Markov Networks

Conditional Random Fields

Planning

Markov Decision Processes

Partially Observable MDP

- Value Iteration
- Approx. Inference

Reinforcement Learning

Applications of AI

Representation

Reasoning Technique

Some of our Grad Courses

522: Artificial Intelligence II : Reasoning and Acting Under Uncertainty

Sample Advanced Topics…..

Relational Reinforcement Learning for Agents in Worlds with Objects, relational learning.

- Probabilistic Relational Learning and Inductive Logic Programming at a Global Scale,

Some of our Grad Courses

503: Computational Linguistics I / Natural Language Processing

Sample Advanced Topics…..

- Topic Modeling (LDA) – Large Scale Graphical Models
- Discourse Parsing by Deep Learning (Neural Nets)
- Abstractive Summarization

Other AI Grad Courses: check them out

532: Topics in Artificial Intelligence (different courses)

540: Machine Learning

505: Image Understanding I: Image Analysis

525: Image Understanding II: Scene Analysis

515: Computational Robotics

Reading Groups

https://www.cs.ubc.ca/cs-research/lci/reading-groups

- Meet regularly to discuss papers and/or topics of interest
- Have email lists to keep people informed
- Designed for faculty/graduate students, but **undergrads are also welcome**
- Excellent way to get to know faculty/grad students in areas of interest