Introduction to

Artificial Intelligence (AI)

Computer Science cpsc502, Lecture 9

Oct, 11, 2011

Slide credit Approx. Inference : S. Thrun, P, Norvig, D. Klein

Today Oct 11

- Bayesian Networks Approx. Inference
- Temporal Probabilistic Models
  - ✓Markov Chains
  - ✓Hidden Markov Models

R&Rsys we'll cover in this course

Problem		Environment: Deterministic	Environment: Stochastic
Static	Constraint Satisfaction	*Vars + Constraints* Arc Consistency SLS Search	
Static	Query	*Logics* → Propositional → First Order → . . . . Search	*Belief Nets* Var. Elimination Approx. Inference Temporal. Inference
Sequential	Planning	*STRIPS* actions precs effects Search	aka influence diagrams *Decision Nets* Var. Elimination *Markov Processes* Value Iteration

Representation

Reasoning Technique

Approximate Inference

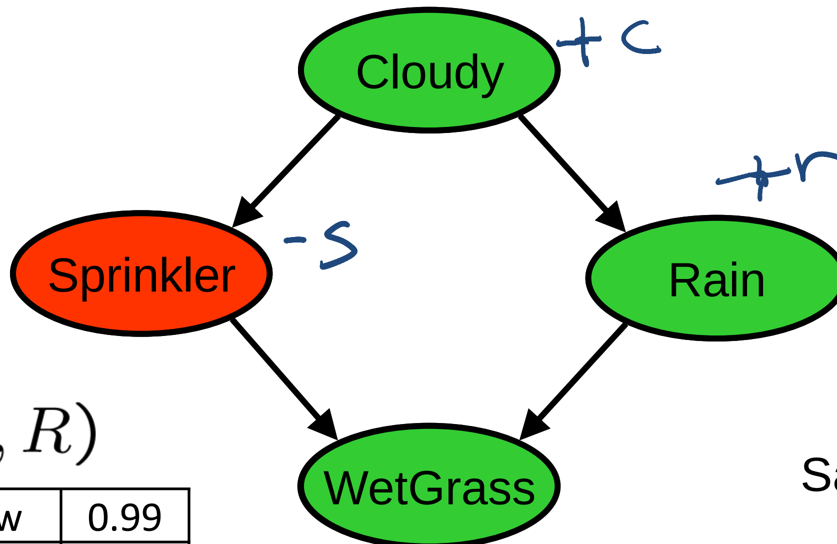
- Basic idea:
  - Draw N samples from a sampling distribution S
  - Compute an approximate probability
  - Show this converges to the true probability P

- Why sample?
  - Inference: getting a sample is faster than computing the right answer (e.g. with variable elimination)

Prior Sampling

$P(C)$

+c	0.5
-c	0.5



$P(S|C)$

+c	+s	0.1
	-s	0.9
-c	+s	0.5
	-s	0.5

$P(R|C)$

+c	+r	0.8
	-r	0.2
-c	+r	0.2
	-r	0.8

$P(W|S,R)$

+s	+r	+w	0.99
		-w	0.01
	-r	+w	0.90
		-w	0.10
-s	+r	+w	0.90
		-w	0.10
	-r	+w	0.01
		-w	0.99

Samples:

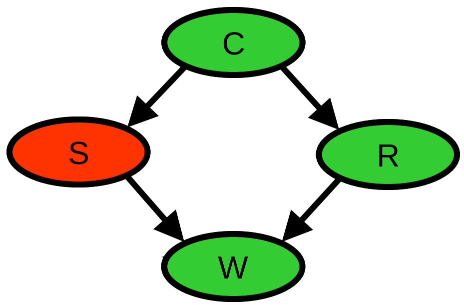
+c, -s, +r, +w

-c, +s, -r, +w

…

Example

- We'll get a bunch of samples from the BN:
  - +c, -s, +r, +w
  - +c, +s, +r, +w
  - ~~-c, +s, +r, -w~~
  - +c, -s, +r, +w
  - -c, -s, -r, +w



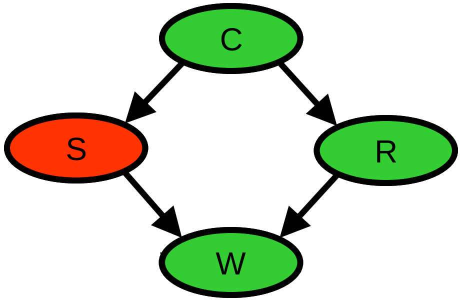
- If we want to know P(W)
  - We have counts <+w:4, -w:1>
  - Normalize to get P(W) = <+w:0.8, -w:0.2>
  - This will get closer to the true distribution with more samples
  - Can estimate anything else, too
  - What about P(C| +w)? P(C| +r, +w)? P(C| -r, -w)?

what's the drawback? Can use fewer samples ?

C+ .75
C- .25

Rejection Sampling

- Let's say we want P(C)
  - No point keeping all samples around
  - Just tally counts of C as we go



- Let's say we want P(C| +s)
  - Same thing: tally C outcomes, but ignore (reject) samples which don't have S=+s
  - This is called rejection sampling
  - It is also consistent for conditional probabilities (i.e., correct in the limit)

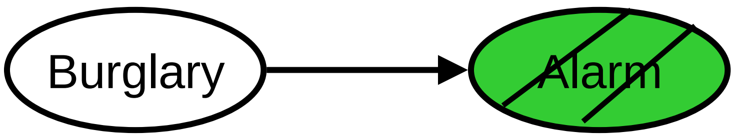
+c, -s, +r, +w
+c, +s, +r, +w
-c, +s, +r, -w
+c, -s, +r, +w
-c, -s, -r, +w

Likelihood Weighting

- Problem with rejection sampling:
  - If evidence is unlikely, you reject a lot of samples
  - You don't exploit your evidence as you sample
  - Consider P(B|+a)

-b, -a
-b, -a
-b, -a
-b, -a
+b, +a

- Idea: fix evidence variables and sample the rest



-b +a
-b, +a
-b, +a
-b, +a
+b, +a

- Problem: sample distribution not consistent!
- Solution: weight by probability of evidence given parents

Likelihood Weighting

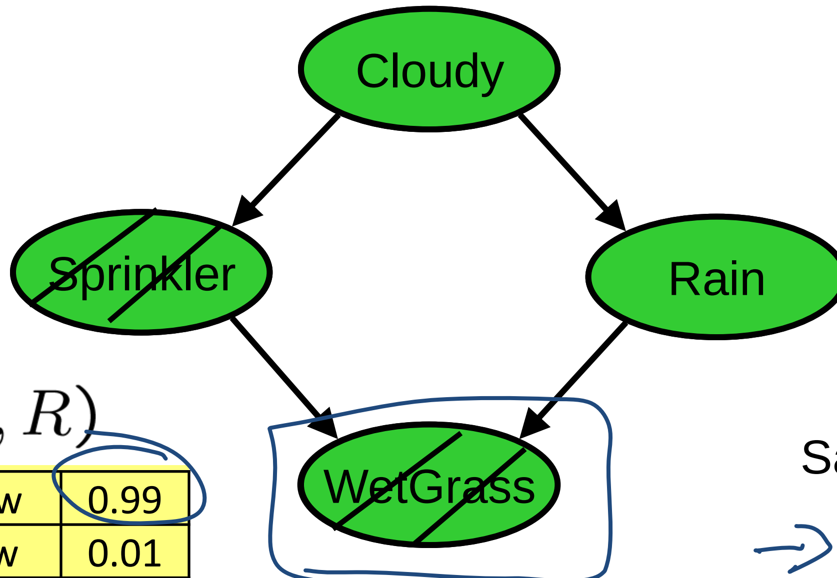
$P(C)$

+c	0.5
-c	0.5

P(R)

$P(S|C)$

+c	+s	0.1
	-s	0.9
-c	+s	0.5
	-s	0.5



$P(R|C)$

+c	+r	0.8
	-r	0.2
-c	+r	0.2
	-r	0.8

$P(W|S,R)$

+s	+r	+w	0.99
		-w	0.01
	-r	+w	0.90
		-w	0.10
-s	+r	+w	0.90
		-w	0.10
	-r	+w	0.01
		-w	0.99

Samples:

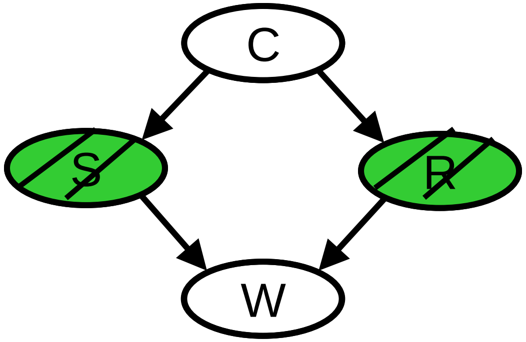
+c, +s, +r, +w

…

$$w = 1.0 \times 0.1 \times 0.99$$

Likelihood Weighting

- Likelihood weighting is good
  - We have taken evidence into account as we generate the sample
  - E.g. here, W's value will get picked based on the evidence values of S, R
  - More of our samples will reflect the state of the world suggested by the evidence

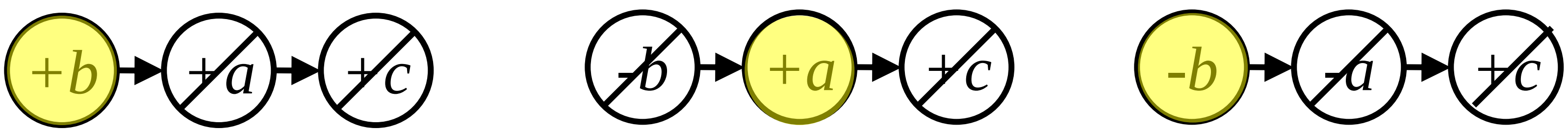


- Likelihood weighting doesn't solve all our problems
  - Evidence influences the choice of downstream variables, but not upstream ones (C isn't more likely to get a value matching the evidence)
- We would like to consider evidence when we sample *every* variable

Markov Chain Monte Carlo

- *Idea*: instead of sampling from scratch, create samples that are each like the last one.

- *Procedure*: resample one variable at a time, conditioned on all the rest, but keep evidence fixed. E.g., for P(b|+c):



- *Properties*: Now samples are not independent (in fact they're nearly identical), but sample averages are still consistent estimators! And can be computed efficiently

→ example of sampling

- *What's the point*: both upstream and downstream variables condition on evidence.

more children

Today Oct 11

- **Bayesian Networks Approx. Inference**
- **Temporal Probabilistic Models**
  - **Markov Chains**
  - **Hidden Markov Models**

Modelling static Environments

So far we have used Bnets to perform inference in **static environments**

- For instance, the system keeps collecting evidence to diagnose the cause of a fault in a system (e.g., a car).

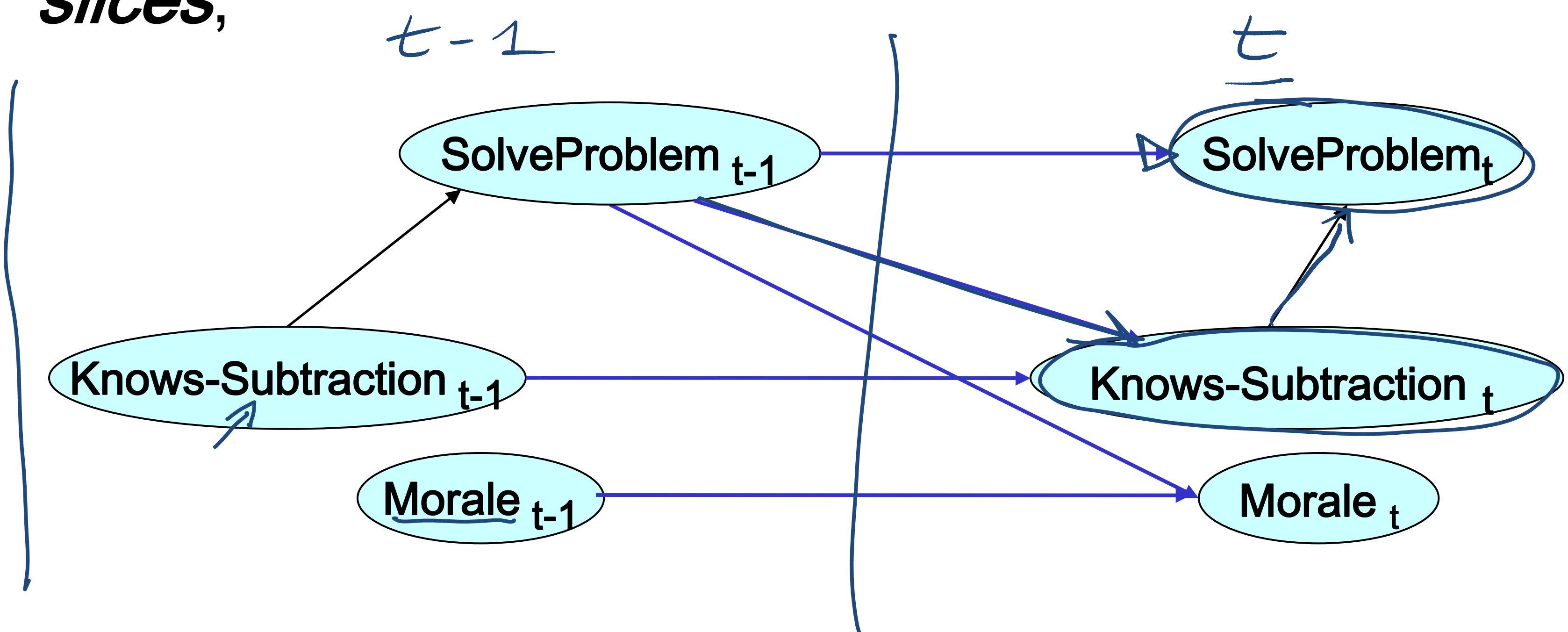
- The environment (values of the evidence, the true cause) does not change as new evidence is gathered

- What does change? *The system's beliefs over possible causes*

Modeling Evolving Environments: Dynamic Bnets

- Often we need to make inferences about evolving environments.

- Represent the state of the world at each specific point in time via a series of snapshots, or ***time slices***,



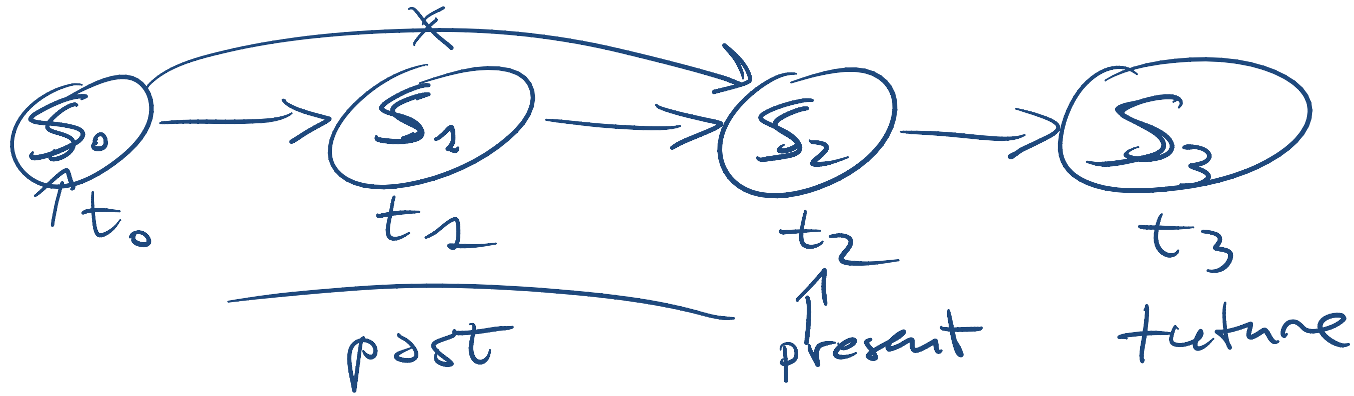
Tutoring system tracing student *knowledge* and *morale*

Today Oct 11

- Bayesian Networks Approx. Inference
- Temporal Probabilistic Models
  - ✓Markov Chains
  - ✓Hidden Markov Models

Simplest Possible DBN

- **One random variable** for each time slice: let's assume S_t represents the state at time t. with domain $\{s_1 \ldots s_n\}$

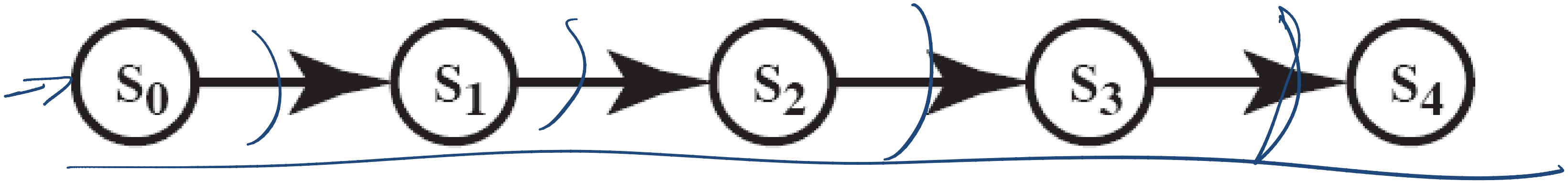


- **Each random variable depends only on the previous one**
- Thus $P(S_{t+1} \mid S_0 \cdots S_t) = P(\underline{S_{t+1}} \mid \underline{S_t})$

- Intuitively S_t conveys all of the information about the history that can affect the future states.
- "The future is independent of the past given the present."

Simplest Possible DBN (cont')

$P(S_4 | S_3)$



- How many CPTs do we need to specify?

4 $P(S_1 | S_0)$ $P(S_2 | S_1)$ etc.

- *Stationary process assumption:* the mechanism that regulates how state variables change overtime is **stationary**, that is it can be described by a single transition model

- $P(S_t|S_{t-1})$ is the same for all t

Stationary Markov Chain (SMC)

A stationary Markov Chain : for all t >0

- $P(S_{t+1}| S_0,\ldots,S_t) = P(S_{t+1}|S_t)$ and Markov assumption
- $P(S_{t+1}|S_t)$ is the same stationary

We only need to specify $P(S_0)$ and $P(S_{t+1}|S_t)$

- Simple Model, easy to specify
- Often the natural model
- The network can extend indefinitely
- **Variations of SMC are at the core of most Natural Language Processing (NLP) applications!**

also used in the PageRank algo (used by Google to rank web pages)

Stationary Markov-Chain: Example

Domain of variable S_i is {t , q, p, a, h, e}

six possible values

We only need to specify…

$P(S_0)$

t	.6
q	.4
p	0
a	0
h	0
e	0

Probability of initial state

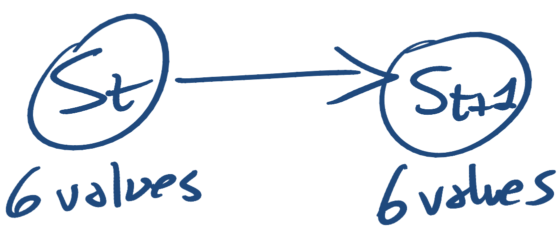
Stochastic Transition Matrix

$P(S_{t+1}|S_t)$

S_t \ S_{t+1}	t	q	p	a	h	e
t	0	.3	0	.3	.4	0
q	.4	0	.6	0	0	0
p	0	0	1	0	0	0
a	0	0	.4	.6	0	0
h	0	0	0	0	0	1
e	1	0	0	0	0	0

← $P(S_{t+1}|S_t = q)$

← $P(S_{t+1}|S_t = p)$



Markov-Chain: Inference

Probability of a sequence of states $S_0 \ldots S_T$

$$P(S_0,\ldots,S_T) = P(S_0)\,P(S_1|S_0)\,P(S_2|S_1)\ldots$$

$$S_0 \to S_1 \to S_2$$

$P(u,e,e) \to$

$P(S_0)$

t	.6
q	.4
p	0
a	0
h	0
e	0

$P(S_{t+1}|S_t)$

	t	q	p	a	h	e
t	0	.3	0	.3	.4	0
q	.4	0	.6	0	0	0
p	0	0	1	0	0	0
a	0	0	.4	.6	0	0
h	0	0	0	0	0	1
e	1	0	0	0	0	0

Example:

$$P(t,q,p) =$$

$$P(t) * P(q|t) * P(p|q)$$

$$.6 * .3 * .6 = .108$$

Key problems in NLP

Noun Verb

"Book me a room near UBC"

w_1 w_2 w_3 w_4 w_5 w_6

$$\boxed{P(w_1,..,w_n)?}$$

Assign a probability to a sentence (a sequence of words)

- Part-of-speech tagging
- Word-sense disambiguation,
- Probabilistic Parsing

Summarization, Machine Translation……..

Predict the next word

$$P(w_n \mid w_1 \ldots \ldots w_{n-1}) = P(w_1 \ldots w_n) / P(w_1 \ldots w_{n-1})$$

- Speech recognition
- Hand-writing recognition
- Augmentative communication for the disabled

$$\boxed{P(w_1,..,w_n)?}$$

Impossible to estimate ☹

$\boxed{P(w_1,..,w_n)?}$ **Impossible to estimate!**

Assuming 10^5 words in Dictionary and average sentence contains 10 words …….

$(10^5)^{10} = 10^{50}$

would contain probabilities

collected from the whole web

Google language repository (22 Sept. 2006) contained "only": 95,119,665,584 sentences

$\sim 10^{11}$

Most sentences will not appear or appear only once ☹

What can we do?

Make a strong simplifying assumption!

Sentences are generated by a Markov Chain

w_1 at the beginning of a sentence

$$P(w_1,..,w_n) = P(w_1 | < S >) \prod_{k=2}^{n} P(w_k \mid w_{k-1})$$

$= P(w_1|<S>)\, P(w_2|w_1)\, P(w_3|w_2) \ldots P(w_k|w_{k-1})$

P(The big red dog barks)=

**P(The|<S>) *** P(big | the) * P(red | big) * ...

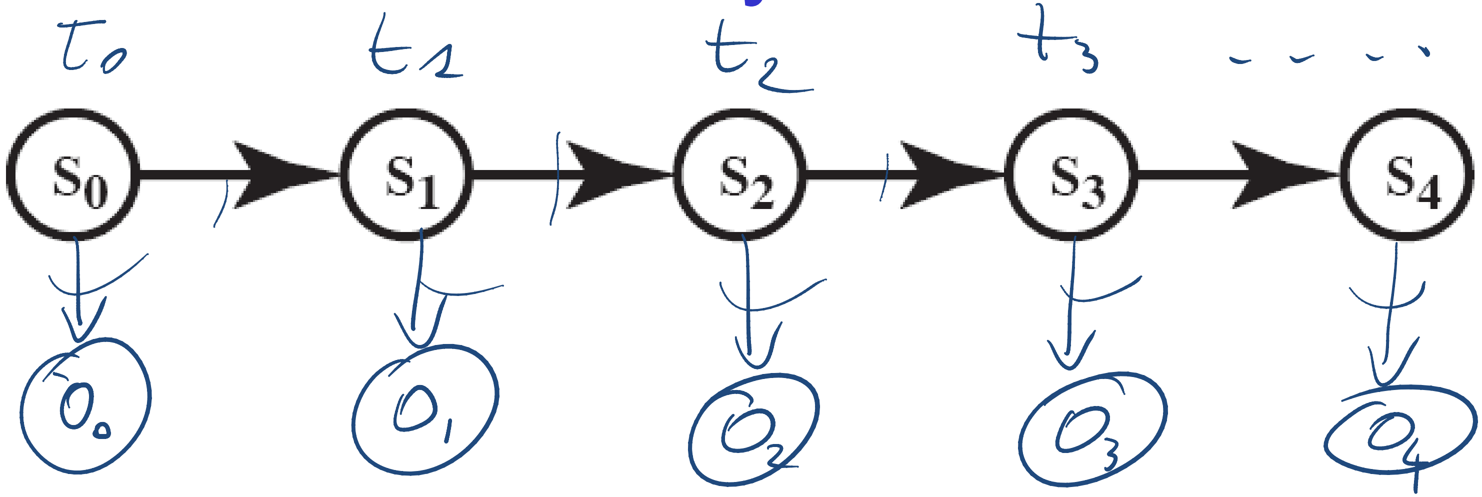
* P(dog | red) * P(barks | dog)

These probs can be assessed in practice!

Today Oct 11

- Bayesian Networks Approx. Inference
- Temporal Probabilistic Models
  - ✓ Markov Chains
  - ✓ (Intro) Hidden Markov Models

How can we minimally extend Markov Chains?



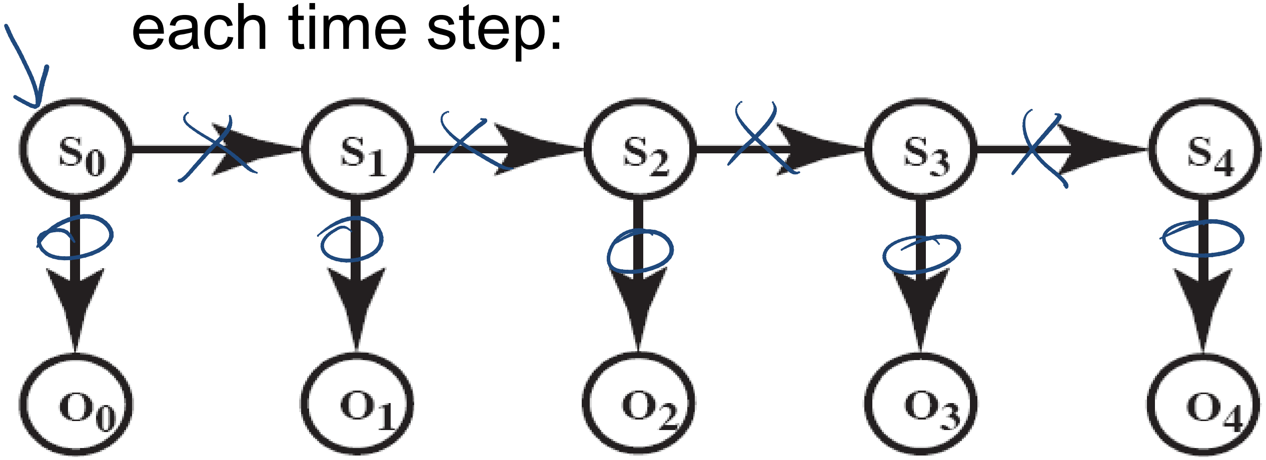
- Maintaining the Markov and stationary assumption

A useful situation to model is the one in which:

- the reasoning system **does not have access** to the states
- but can **make observations** that give some information about the current state

Hidden Markov Model

- A Hidden Markov Model (HMM) starts with a Markov chain, and adds a noisy observation about the state at each time step:

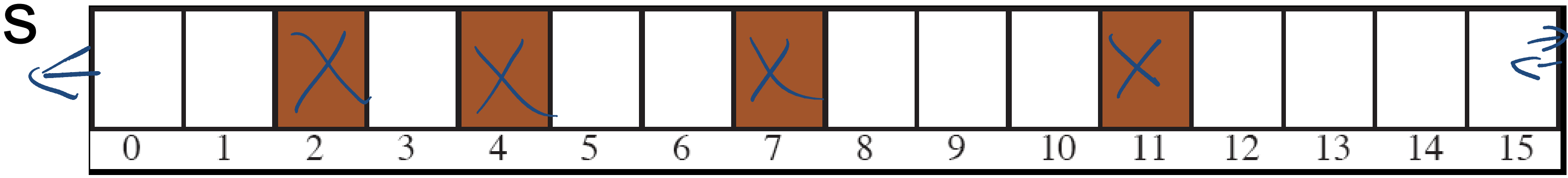


- |domain(S)| = *k*
- |domain(O)| = *h*

- $P(S_0)$ specifies initial conditions K
- $P(S_{t+1}|S_t)$ specifies the dynamics K×K
- $P(O_t|S_t)$ specifies the sensor model K×h {K prob. dist. over O}

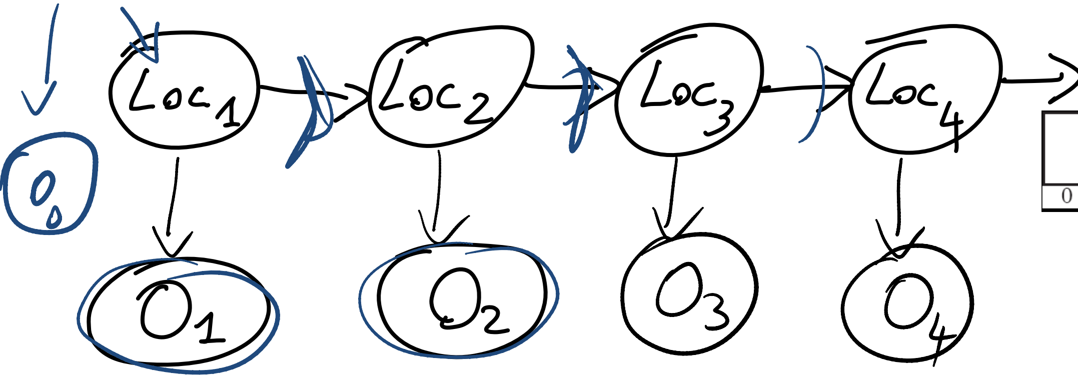
Example: Localization for “Pushed around” Robot

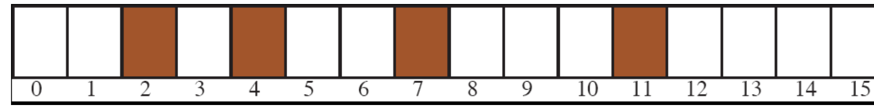
- **Localization** (where am I?) is a fundamental problem in robotics
- Suppose a robot is in a circular corridor with 16 locations



  - There are four doors at positions: 2, 4, 7, 11
  - The Robot initially doesn’t know where it is
  - The Robot is pushed around. After a push it can stay in the same location, move left or right.
  - The Robot has Noisy sensor telling whether it is in front of a door

This scenario can be represented as…





- **Example Stochastic Dynamics**: when pushed, it stays in the same location p=0.2, moves left or right with equal probability .4

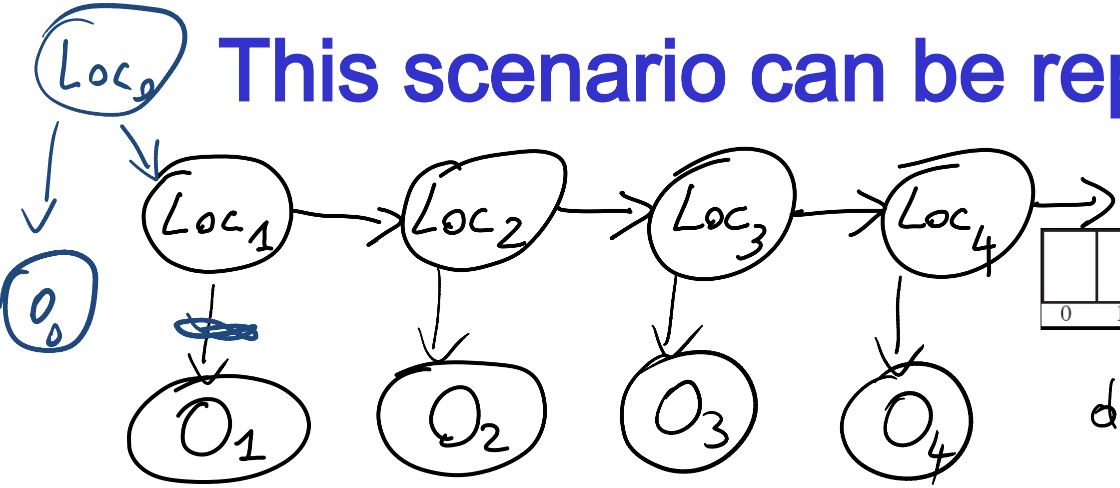
$P(Loc_{t+1} / Loc_t)$

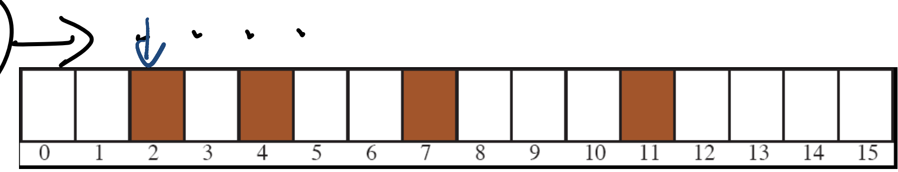
Loc_t \ Loc_{t+1}	0	1	2	……	15
0	.2	.4	0	…… 0	.4
1	.4	.2	.4	0 ……	0
2					
3					
⋮					
15					

$P(Loc_1)$ =

0	1	2	…	15
1/16	1/16	1/16	— — —	

This scenario can be represented as…





$dom(Loc_i) = \{0, 1, \ldots, 15\}$

Example of Noisy sensor telling whether it is in front of a door.

- If it is in front of a door $P(O_t = T) = .8$
- If not in front of a door $P(O_t = T) = .1$

$\boxed{P(O_t \mid Loc_t)}$

Loc_t	$P(O_t=T)$	$P(O_t=F)$
0	.1	.9
1	.1	.9
2	.8	.2
3	.1	.9
4	.8	.2
⋮		

16 prob. distributions

Useful inference in HMMs

- **Localization**: Robot starts at an unknown location and it is pushed around *t* times. It wants to determine where it is

$$P(Loc_t \mid o_0, o_1, \ldots, o_t)$$

- **In general (Filtering):** compute the posterior distribution over the current state given all evidence to date

$$P(X_t \,/\, o_{0:t}) \quad or \quad P(X_t \,/\, e_{0:t})$$

Other HMM Inferences (next time)

- **Smoothing** (posterior distribution over a *past* state given all evidence to date)

$$P(X_k / e_{0:t}) \text{ for } 1 \leq k < t$$

- **Most Likely Sequence** (given the evidence seen so far)

$$\text{argmax}_{x_{0:t}} P(X_{0:t} \mid e_{0:t})$$

TODO for this Thurs

- Work on Assignment2

- Study the Handout (on approx. inference)

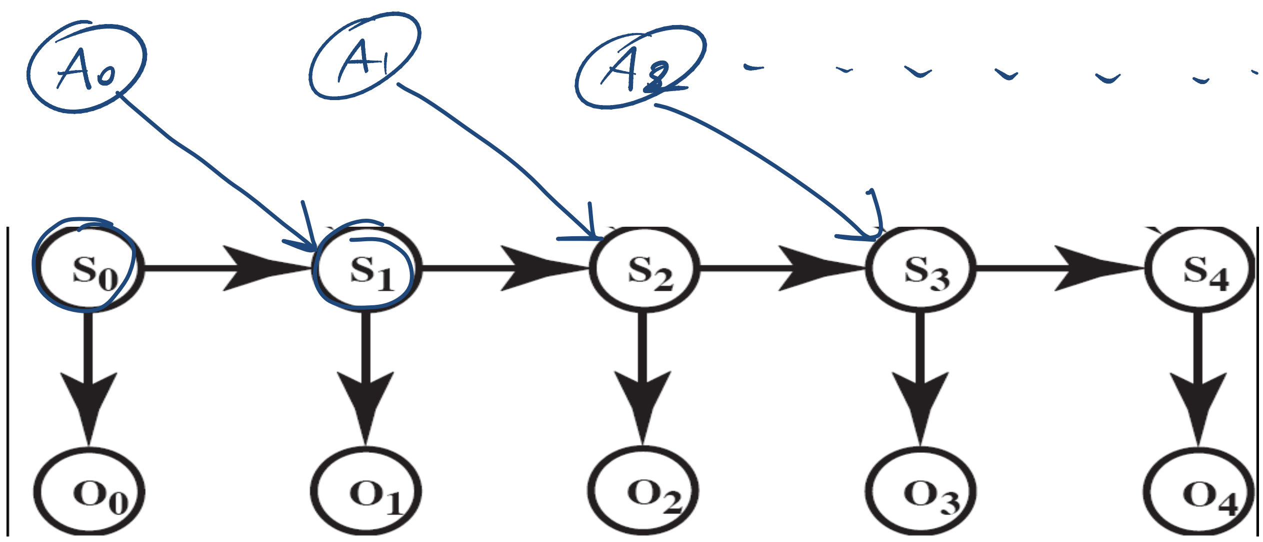
Available outside my office after 1pm

Also Do exercise 6.E (parts on importance sampling and particle filtering are optional)

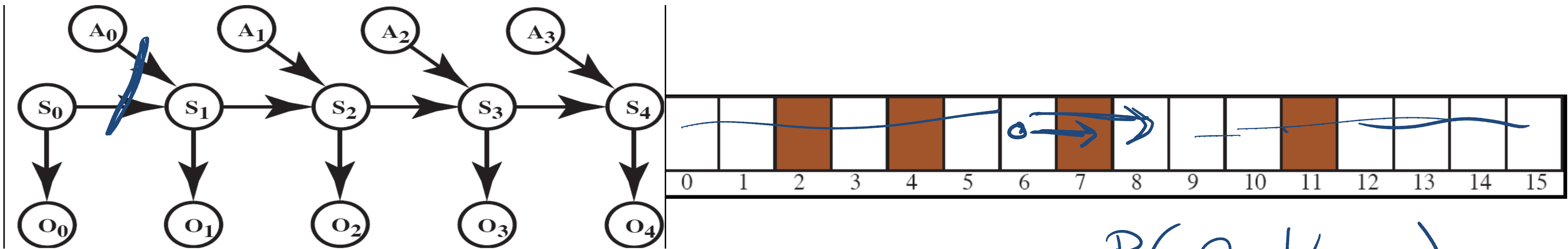
http://www.aispace.org/exercises.shtml

Example : Robot Localization

- Suppose a robot wants to determine its location based on its actions and its sensor readings
- Three actions: *goRight, goLeft, Stay*
- This can be represented by an augmented HMM



Robot Localization Sensor and Dynamics Model



$P(O_t \mid Loc_t)$

- **Sample Sensor Model** (assume same as for pushed around)
- **Sample Stochastic Dynamics**: $P(Loc_{t+1} \mid Action_t, Loc_t)$

$P(Loc_{t+1} = L \mid Action_t = goRight, Loc_t = L) = 0.1$

$P(Loc_{t+1} = L+1 \mid Action_t = goRight, Loc_t = L) = 0.8$

$P(Loc_{t+1} = L+2 \mid Action_t = goRight, Loc_t = L) = 0.074$

$P(Loc_{t+1} = L' \mid Action_t = goRight, Loc_t = L) = \boxed{0.002}$ *for all other locations L'*

x13

- All location arithmetic is modulo 16
- The action *goLeft* works the same but to the left

Dynamics Model More Details

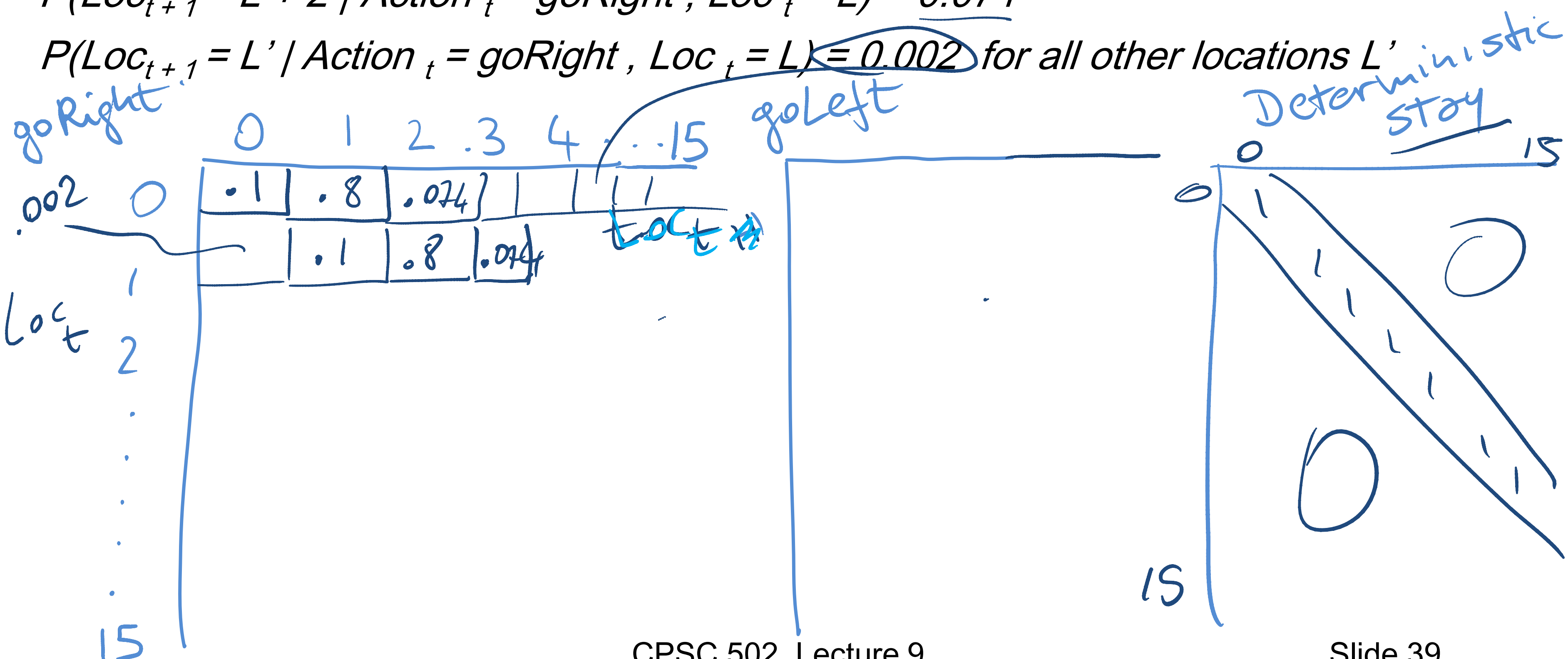
- **Sample Stochastic Dynamics:** $P(Loc_{t+1} \mid Action, Loc_t)$

$P(Loc_{t+1} = L \mid Action_t = goRight, Loc_t = L) = 0.1$

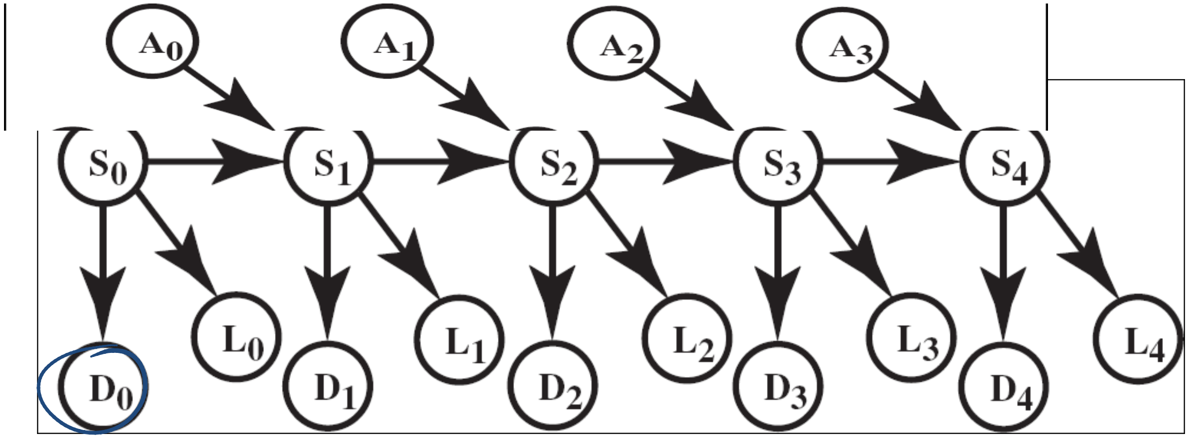
$P(Loc_{t+1} = L+1 \mid Action_t = goRight, Loc_t = L) = 0.8$

$P(Loc_{t+1} = L + 2 \mid Action_t = goRight, Loc_t = L) = 0.074$

$P(Loc_{t+1} = L' \mid Action_t = goRight, Loc_t = L) = 0.002$ for all other locations L'



Robot Localization additional sensor



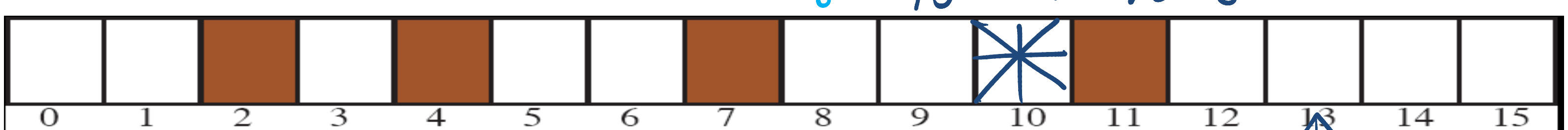
$L_t = T$

the Robot senses light

- **Additional Light Sensor:** there is light coming through an opening at location 10 $P(L_t / Loc_t)$

$P(L_t = F)$ .2 .05 .01 .05 .2 .4

$P(L_t = T)$ .8 .95 .99 .95 .8 .6



- **Info from the two sensors is combined :"Sensor Fusion"**

The Robot starts at an unknown location and must determine where it is

The model appears to be too ambiguous

- Sensors are too noisy
- Dynamics are too stochastic to infer anything

But inference actually works pretty well.

Let's check:

`http://www.cs.ubc.ca/spider/poole/demos/localization/localization.html`

You can use standard Bnet inference. However you typically take advantage of the fact that time moves forward (not in 322)

Sample scenario to explore in demo

- Keep making observations without moving. What happens?
- Then keep moving without making observations. What happens?
- Assume you are at a certain position alternate moves and observations
- ….

HMMs have many other applications….

Natural Language Processing: e.g., Speech Recognition

- *States:* phoneme \ word

HMM 1 HMM2

My name is

- *Observations*: $\boxed{\text{acoustic signal}}$ \ phoneme

Bioinformatics: Gene Finding

- *States:* coding / non-coding region × × ✓ ✓ ✓ × ×
- *Observations:* DNA Sequences → A T C G G A A

For these problems the critical inference is:

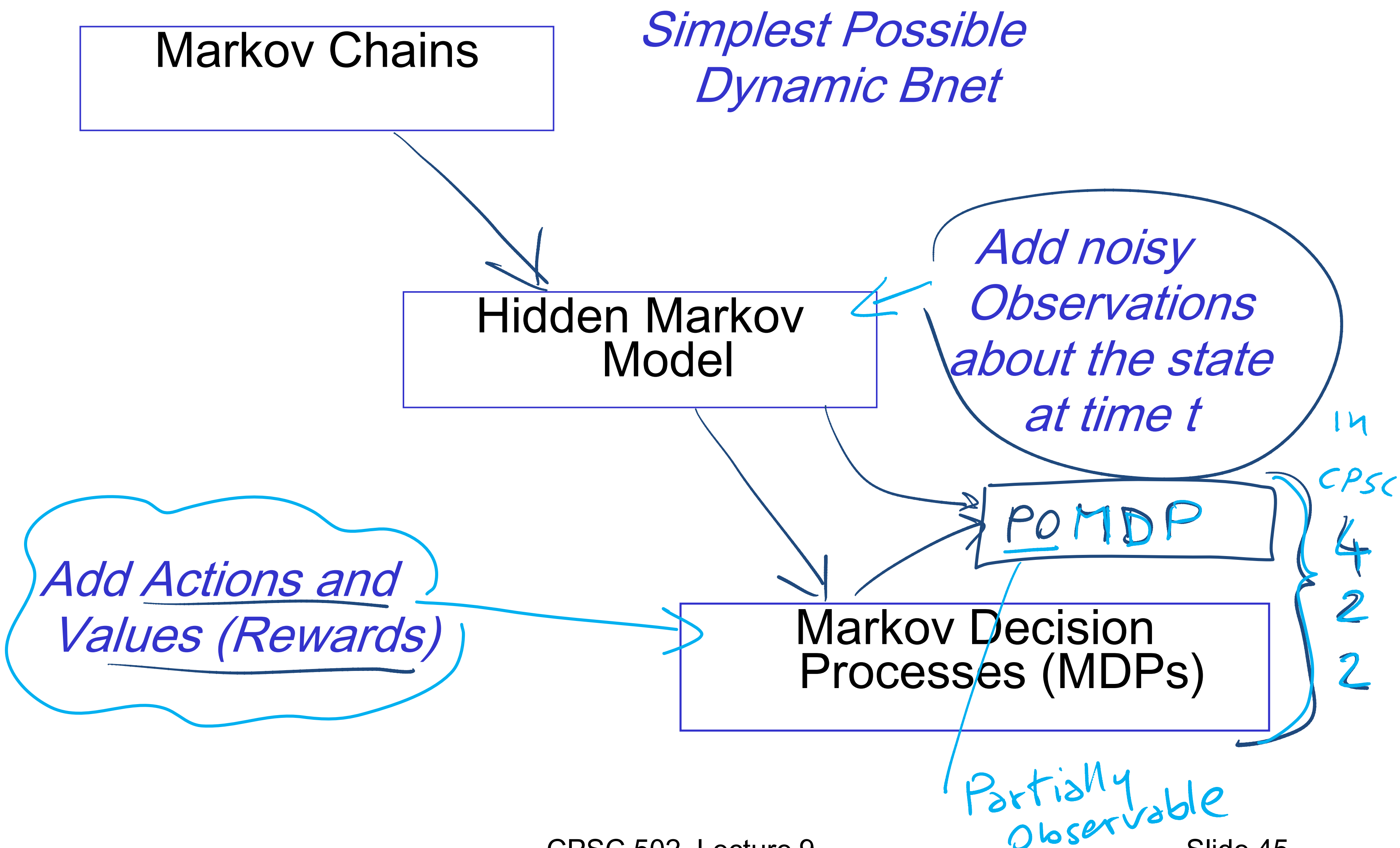
find the most likely sequence of states given a sequence of observations

Viterbi Algo

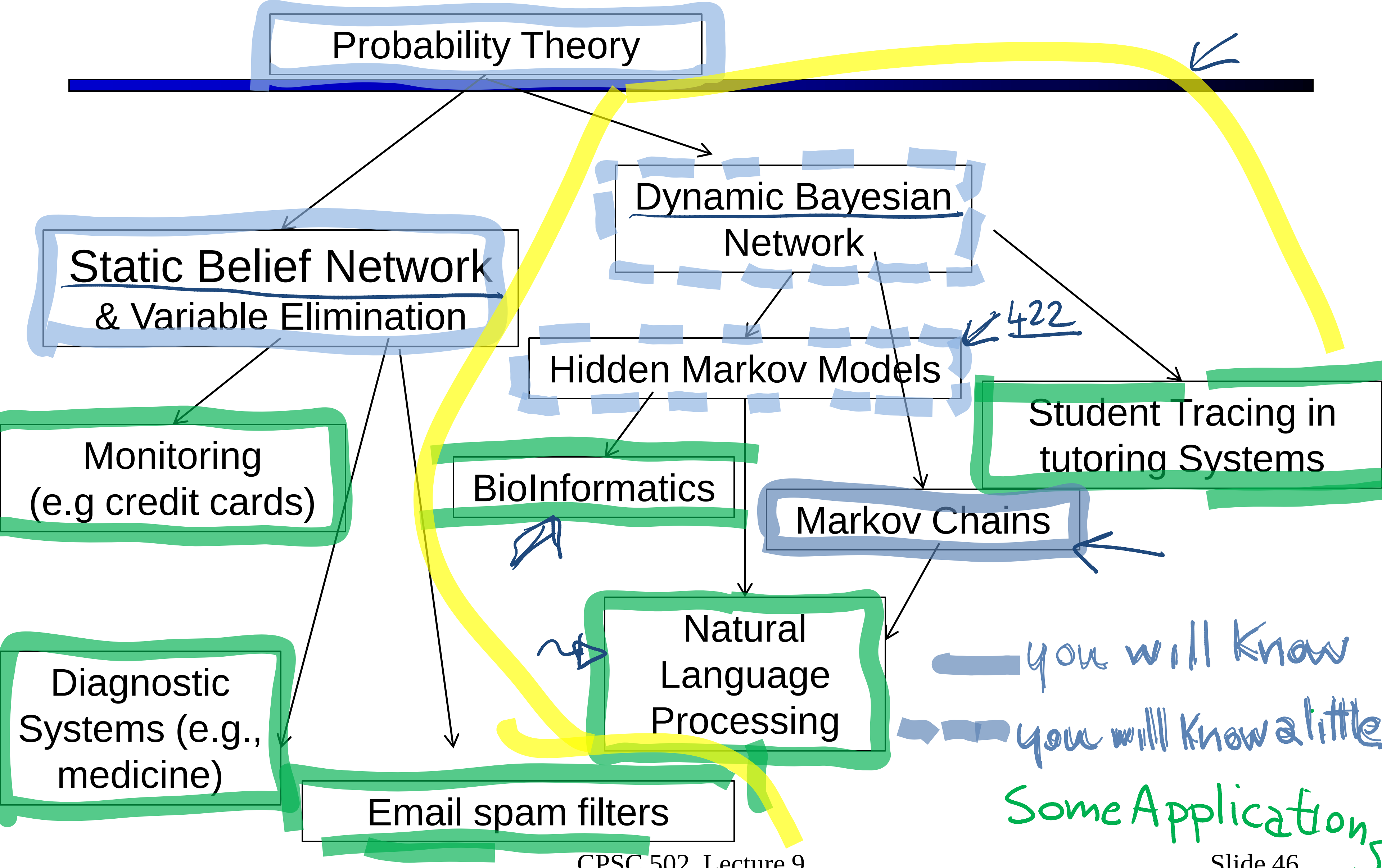
NEED to explain Filtering

Because it will be used in POMDPs

Markov Models



Answering Query under Uncertainty



Lecture Overview

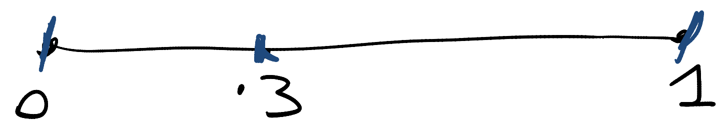
- **Recap**
- **Temporal Probabilistic Models**
- Start Markov Models
  - Markov Chain
  - Markov Chains in Natural Language Processing

Sampling a discrete probability distribution

e.g. Sim. Annealing. Select n' with probability P

P = .3

generate random number in [0,1]



If < .3 accept n'

e.g. Beam Search: Select K individuals. Probability of selection proportional to their value

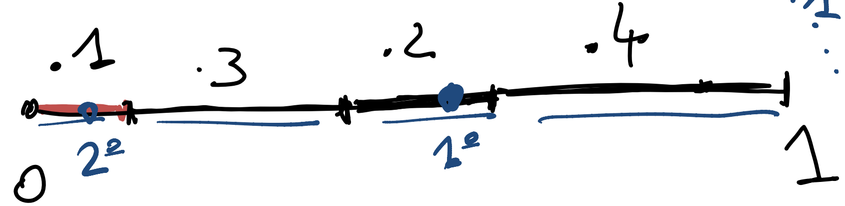
SAME HERE

→ n_1 $\boxed{P_1 = .1}$

→ n_2 $\boxed{P_2 = .3}$

→ n_3 $\boxed{P_3 = .2}$

→ n_4 $\boxed{P_4 = .4}$

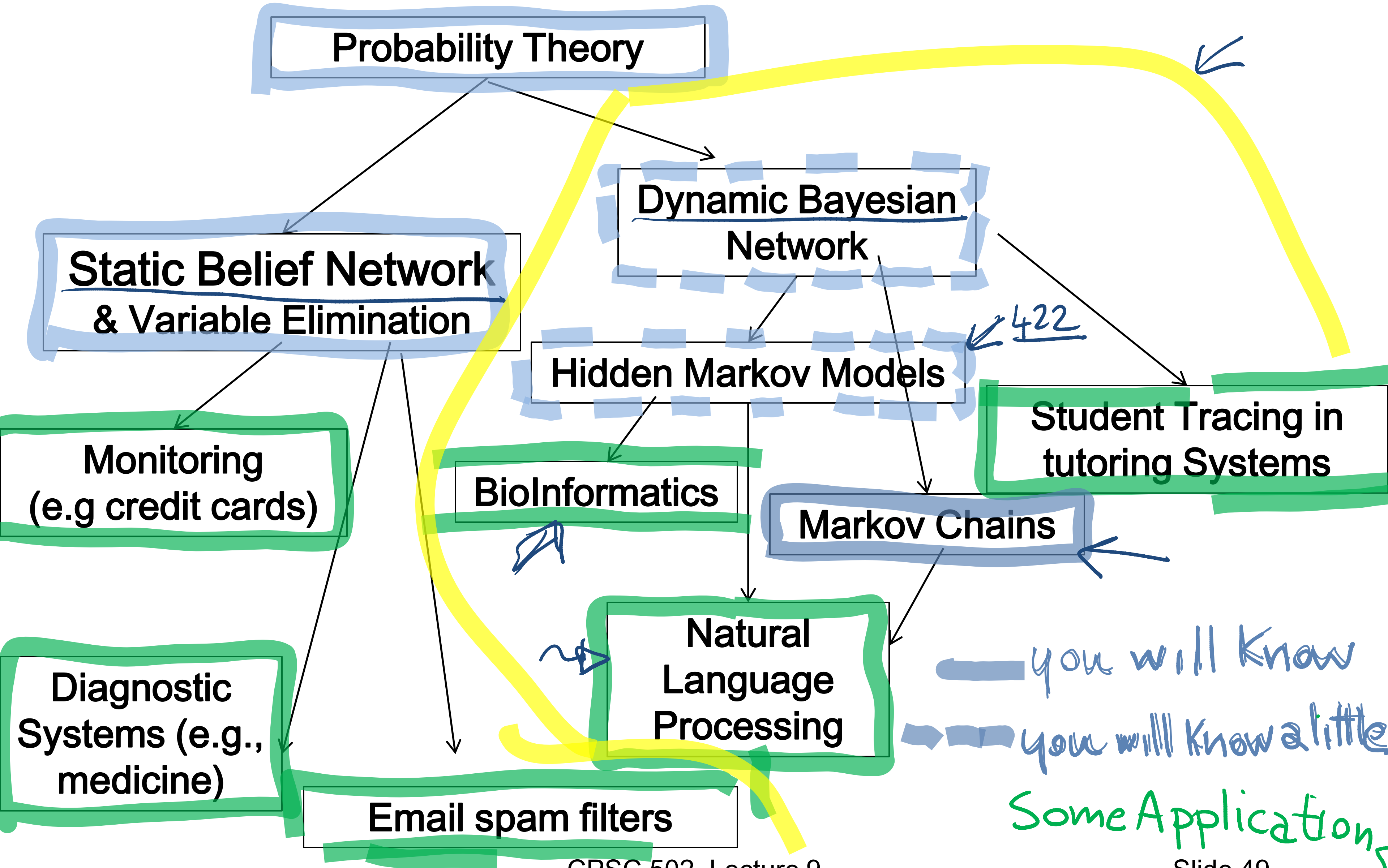


n_3 first sample

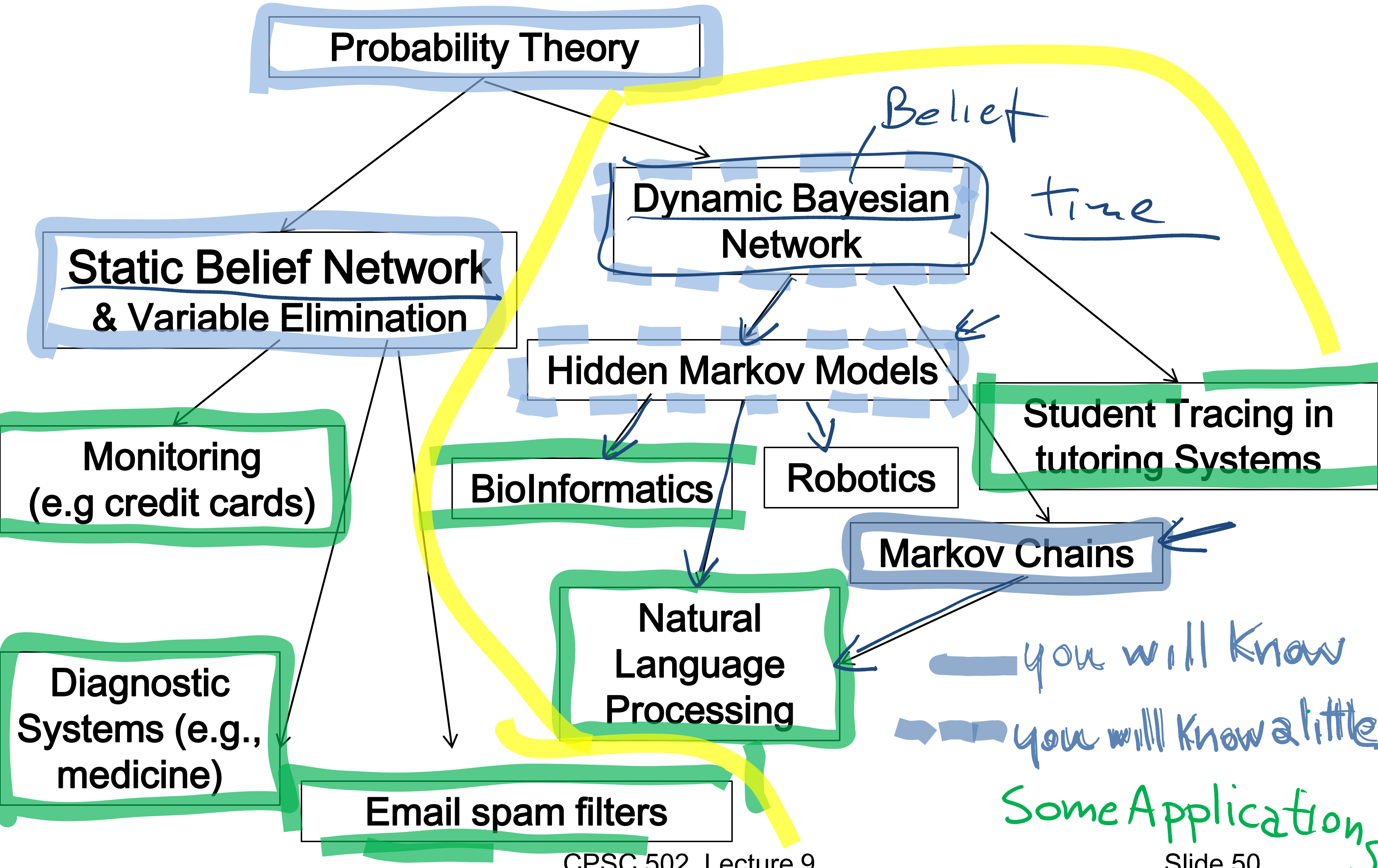
n_1 second sample

⋮

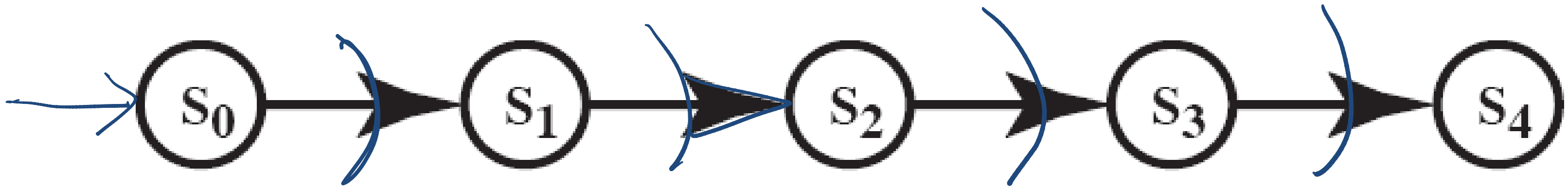
Answering Query under Uncertainty



Answering Queries under Uncertainty



Stationary Markov Chain (SMC)



A stationary Markov Chain : for all t >0 $\quad |dom(S_i)| = K$

- $P(S_{t+1} | S_0,\ldots,S_t) = P(S_{t+1} | S_t)$ and
- $P(S_{t+1} | S_t)$ the same $\forall t$

We only need to specify $\underline{P(S_0)}$ K and $\underline{P(S_{t+1} | S_t)}$

$K \times K$

- Simple Model, easy to specify
- Often the natural model
- The network can extend indefinitely
- **Variations of SMC are at the core of most Natural Language Processing (NLP) applications!**

K prob distrib.