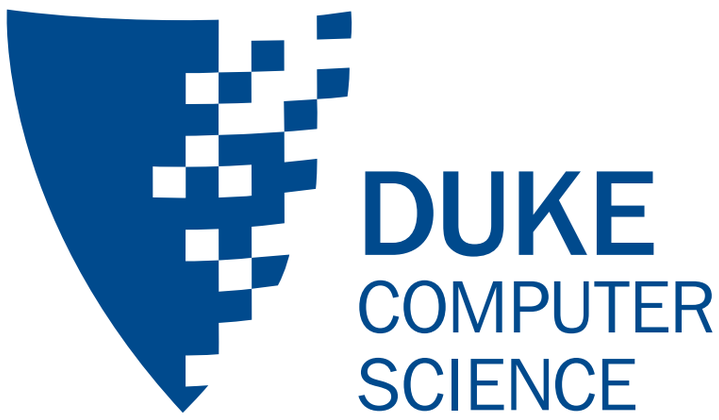
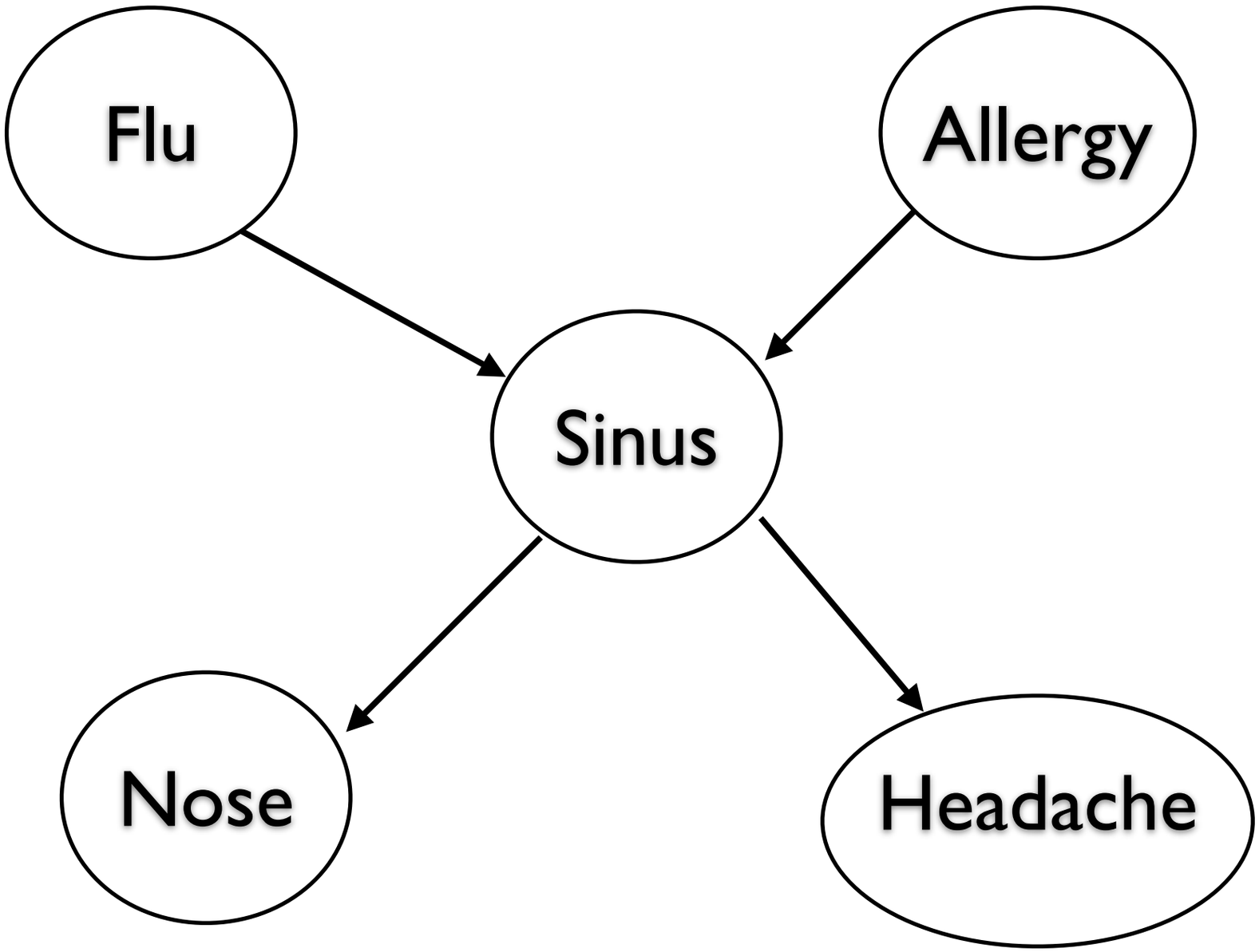
Hidden Markov Models

George Konidaris
gdk@cs.duke.edu

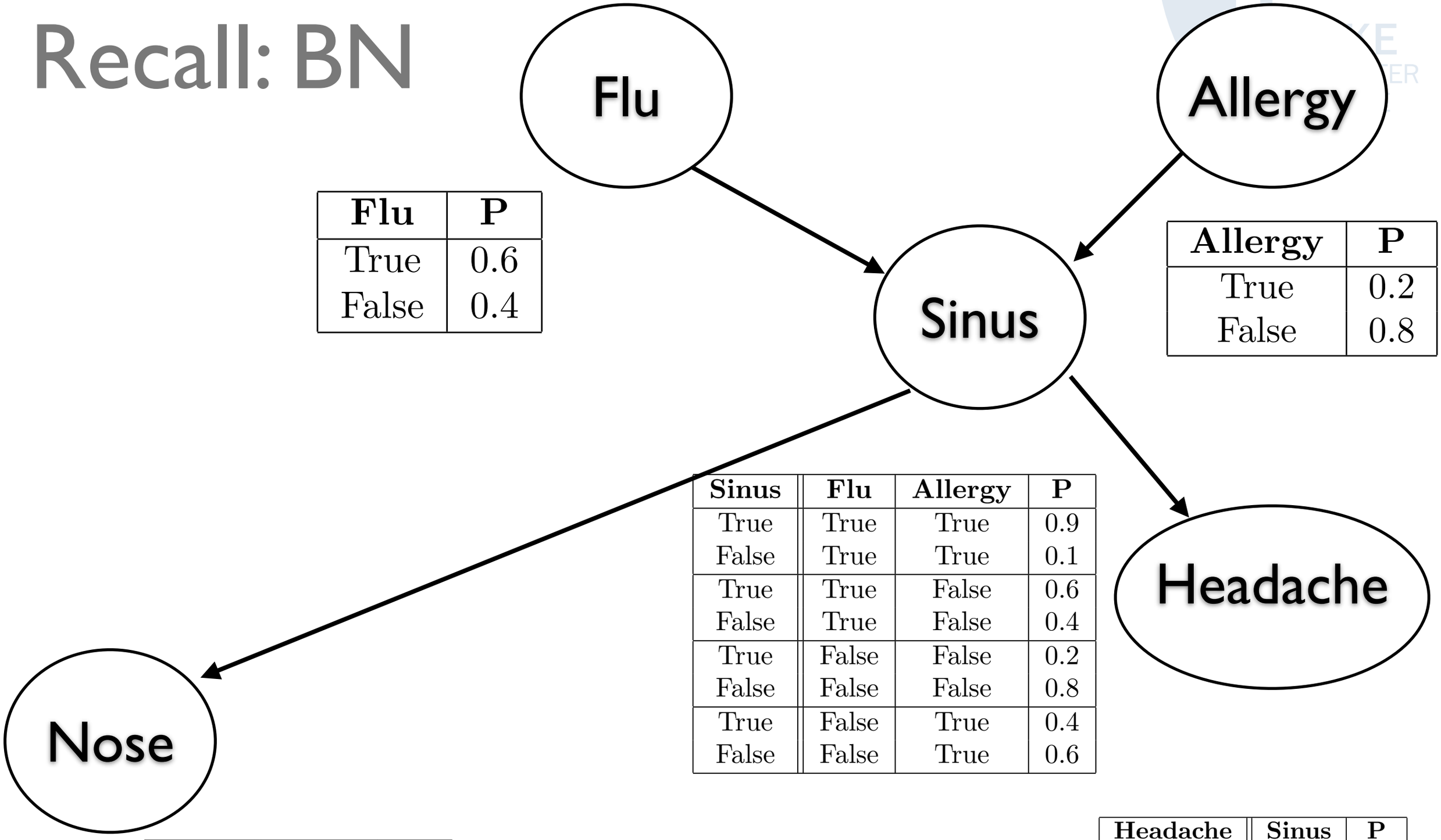


Spring 2016

Recall: Bayesian Network



Recall: BN



Flu	P
True	0.6
False	0.4

Allergy	P
True	0.2
False	0.8

Sinus	Flu	Allergy	P
True	True	True	0.9
False	True	True	0.1
True	True	False	0.6
False	True	False	0.4
True	False	False	0.2
False	False	False	0.8
True	False	True	0.4
False	False	True	0.6

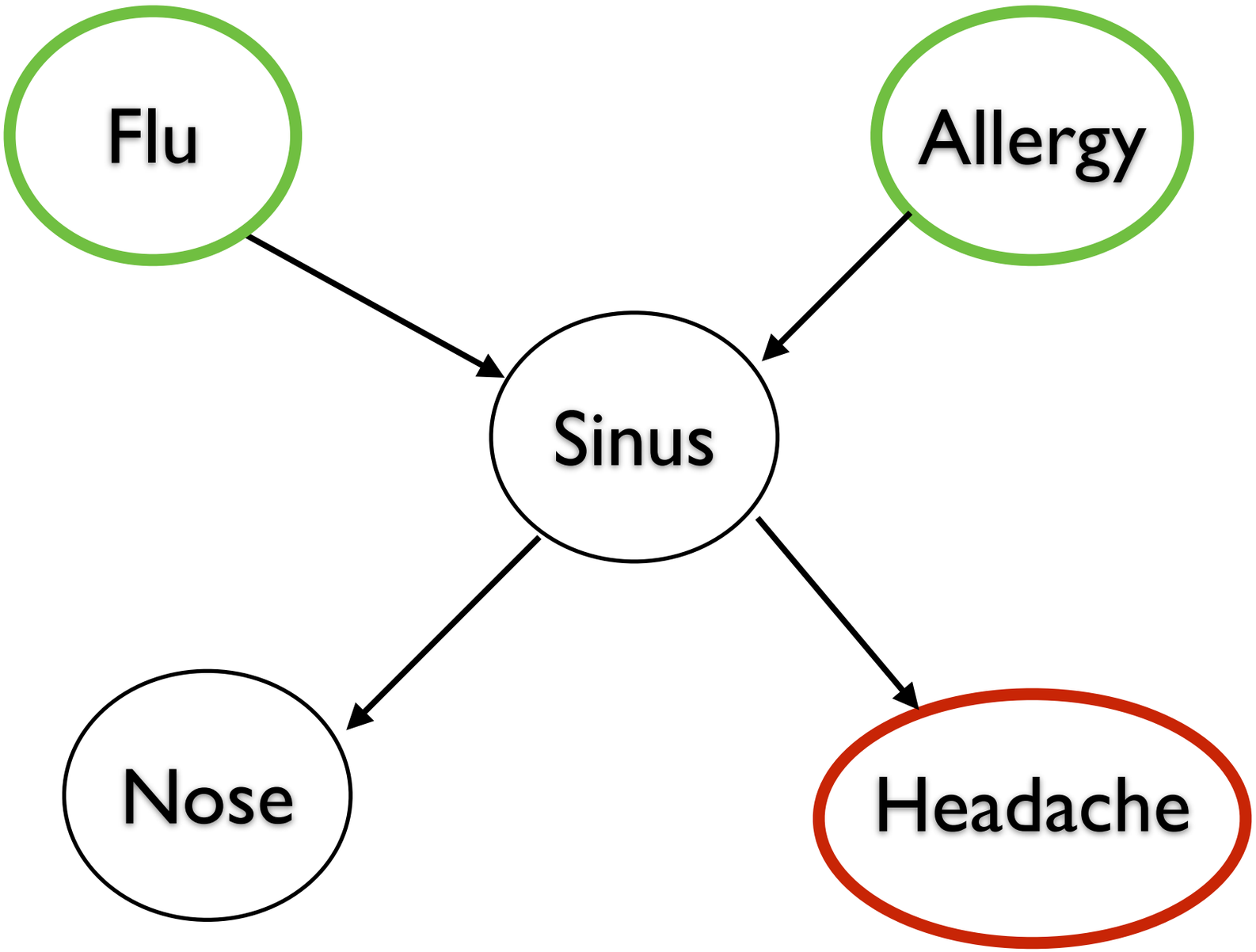
Nose	Sinus	P
True	True	0.8
False	True	0.2
True	False	0.3
False	False	0.7

Headache	Sinus	P
True	True	0.6
False	True	0.4
True	False	0.5
False	False	0.5

joint: 32 (31) entries

Inference

Given A compute P(B | A).



Time

Bayesian Networks (so far) contain no notion of ***time***.

However, in many applications:

- Target tracking
- Patient monitoring
- Speech recognition
- Gesture recognition

… how a signal changes over time is critical.

States

In probability theory, we talked about *atomic events:*

- All possible outcomes.
- Mutually exclusive.

In time series, we have **state**:

- System is in a **state** at time t.
- Describes system completely.
- Over time, transition from *state to state*.

Example

The weather today can be:

- Hot
- Cold
- Chilly
- Freezing

The weather has four *states*.

At each ***point in time***, the system is in ***one (and only one) state***.

The Markov Assumption

We are probabilistic modelers, so we'd like to model:

$$P(S_t|S_{t-1}, S_{t-2}, \ldots, S_0)$$

A state has the Markov property when we can write this as:

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

Special kind of independence assumption:

- *Future independent of past given present.*

Markov Assumption

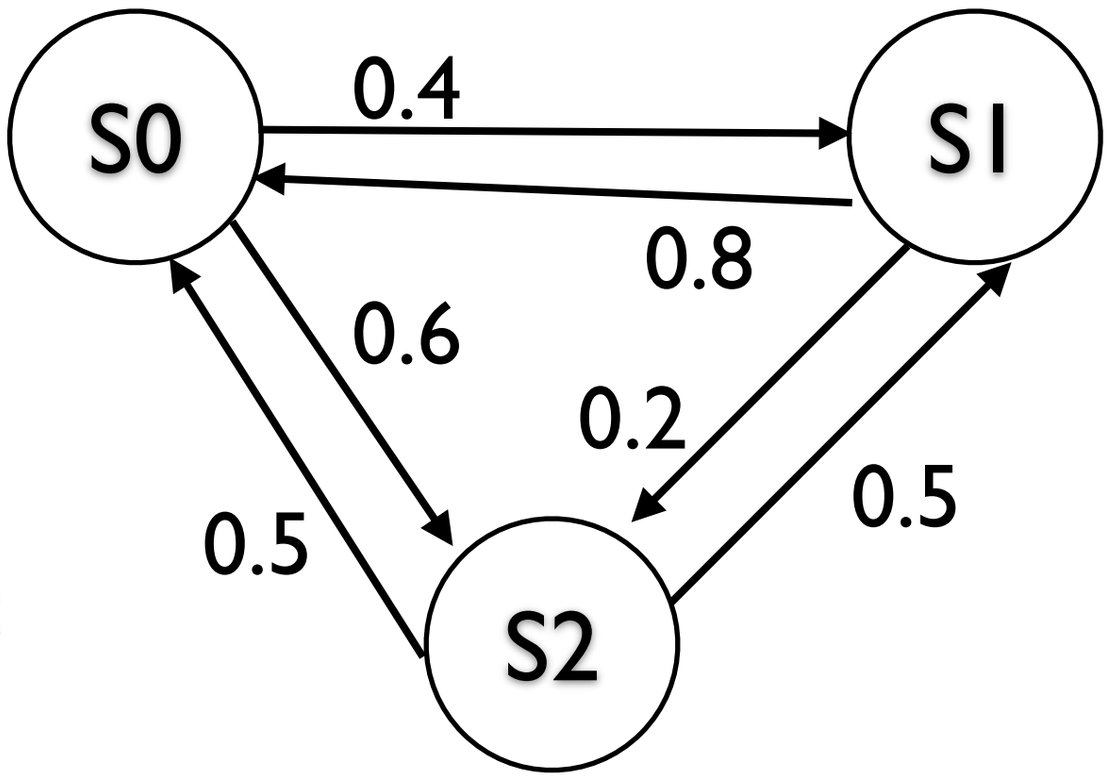
Model that has it is a ***Markov model***.

Sequence of states thus generated is a ***Markov chain***.

Can describe transition probabilities with matrix:

- $P(S_i \mid S_j)$
- Steady state probabilities.
- Convergence rates.

State Machines



P(S0 | S1) = 0.8
P(S0 | S2) = 0.5
P(S1 | S0) = 0.4
P(S1 | S2) = 0.5
P(S2 | S0) = 0.6
P(S2 | S1) = 0.2

Time implicit

states not
state vars!

State Machines

Assumptions:

- Markov assumption.
- Transition probabilities don't change with time.
- Event space doesn't change with time.
- Time moves in discrete increments.

Hidden State

State machines are cool but:

- Often state is not observed directly.
- State is latent, or hidden.

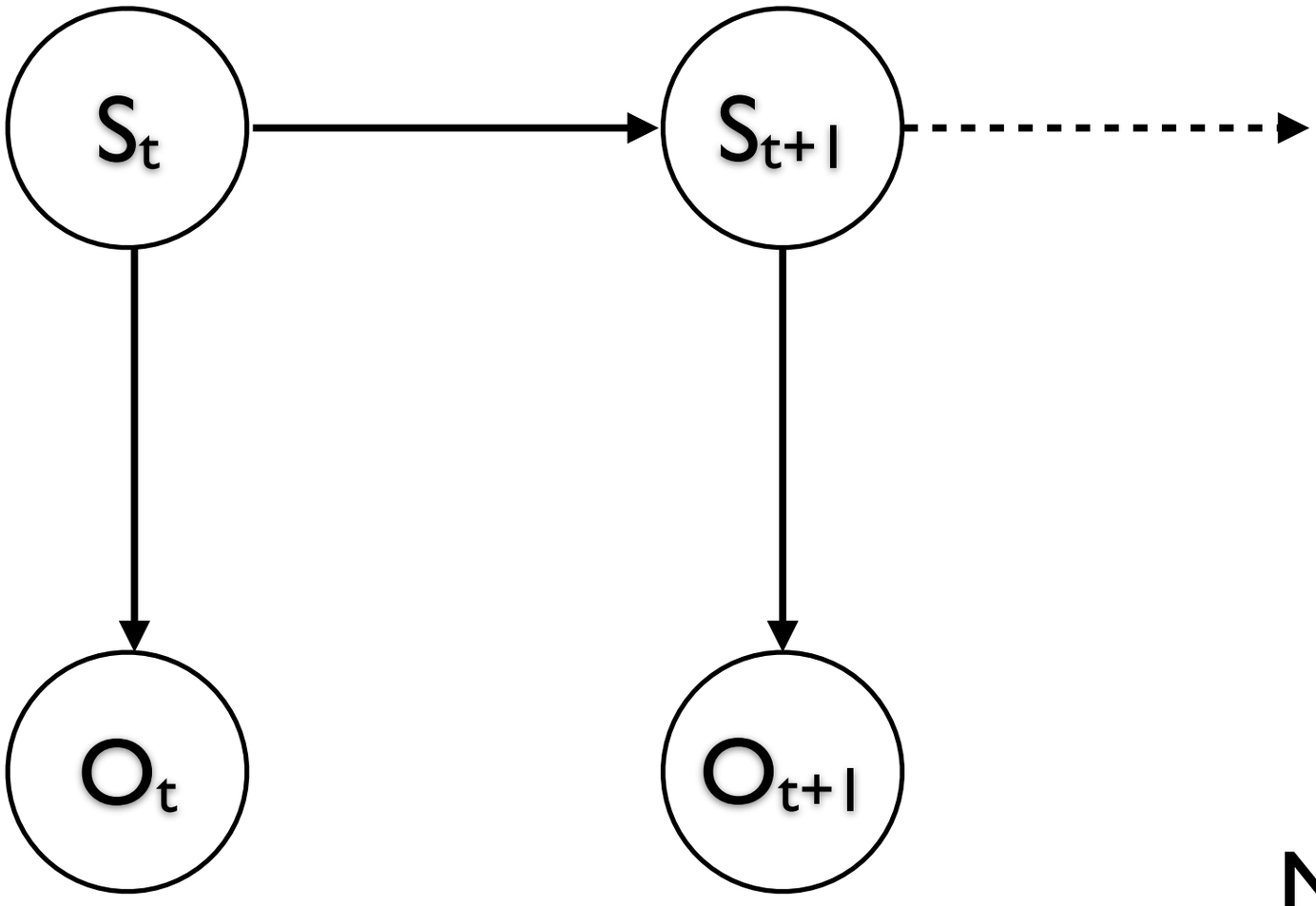
State:
forehand

Instead you see an *observation*, which contains information about the hidden state.

Examples

State	Observation
Word	Phoneme
Chemical State	Color, Smell etc.
Flu?	Runny Nose
Cardiac Arrest?	Pulse

Hidden Markov Models



Must store:

- $P(O \mid S)$
- $P(S_{t+1} \mid S_t)$

HMMs

Monitoring/Filtering

- $P(S_t \mid E_0 \ldots E_t)$
- E.g., estimate patient disease state.

Prediction

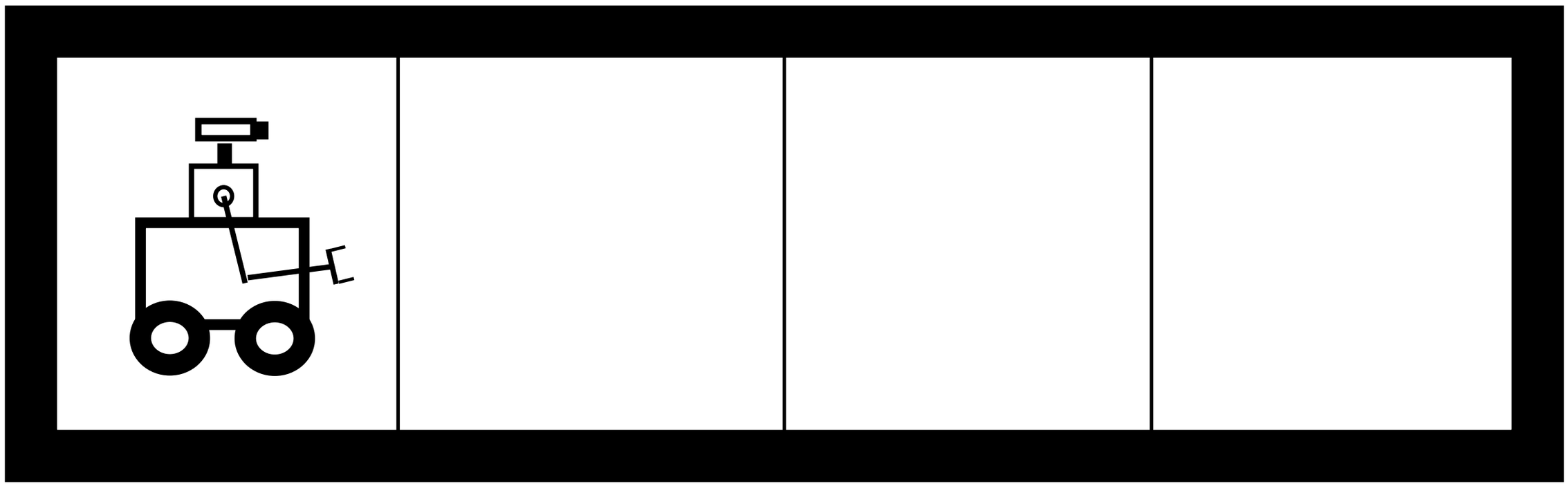
- $P(S_t \mid E_0 \ldots E_k)$, $k < t$.
- Given first two phonemes, what word?

Smoothing

- $P(S_t \mid E_0 \ldots E_k)$, $k > t$
- What happened back there?

Most likely state sequence. (a path, not a distribution)

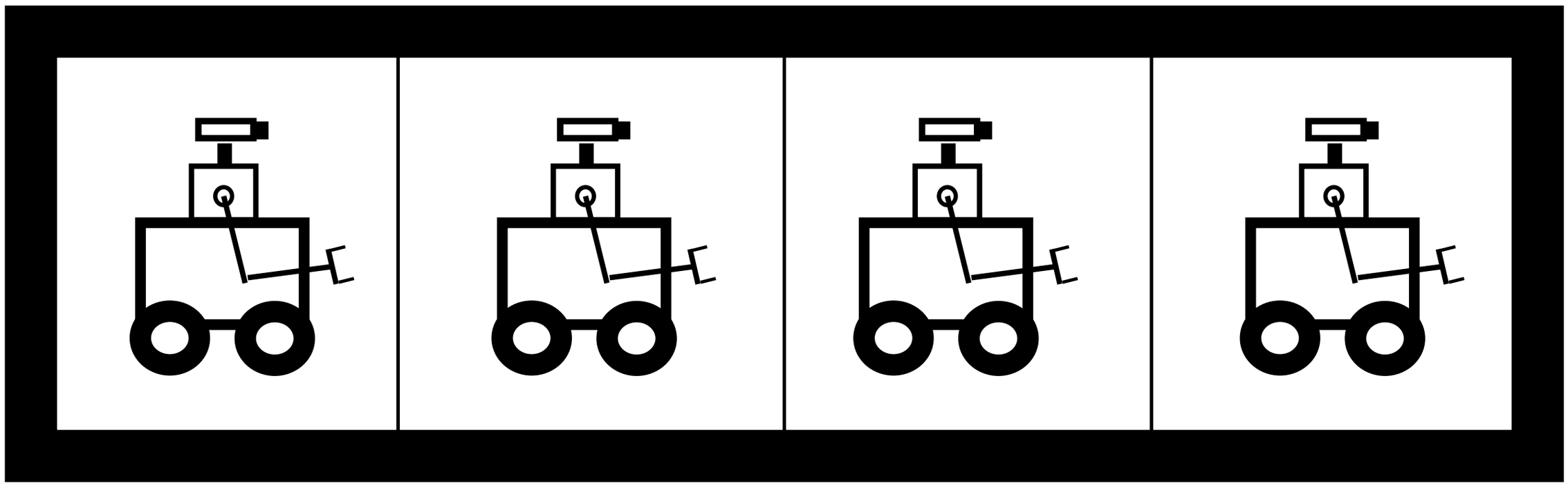
Example: Robot Localization



observations:
walls each side?

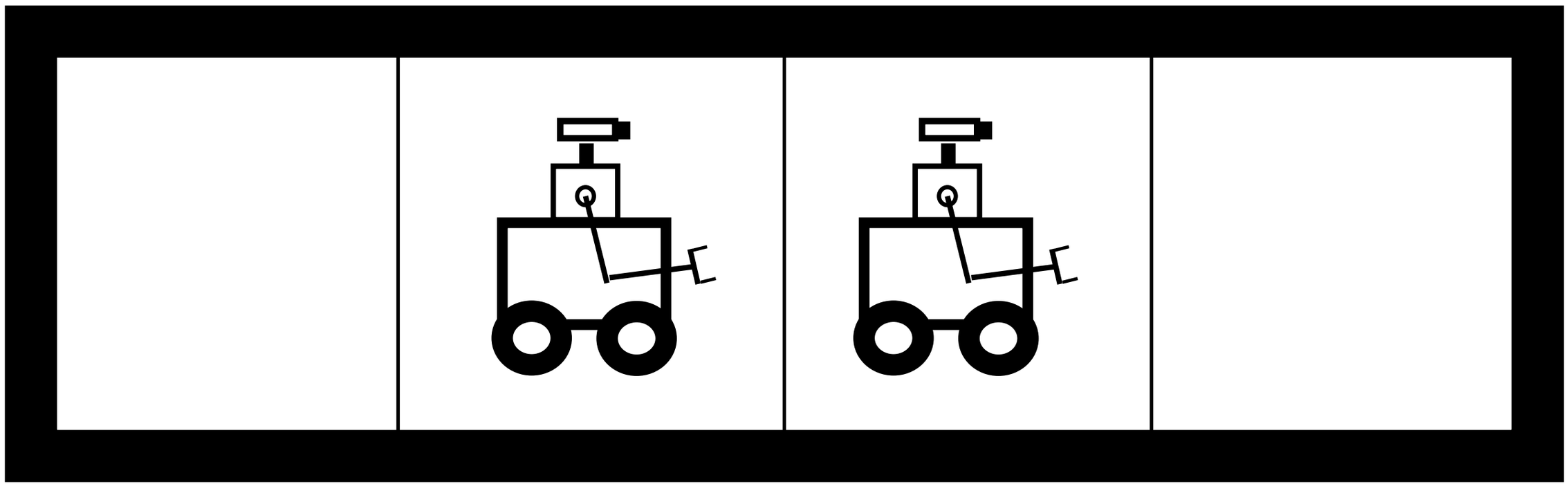
states:
position

Example: Robot Localization



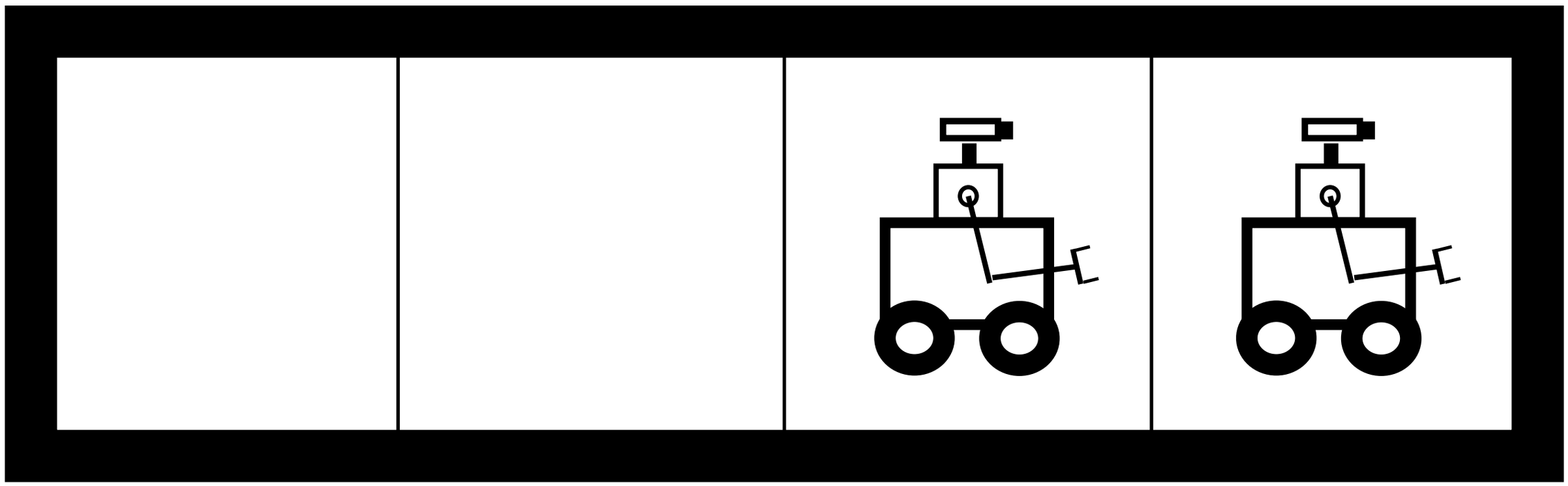
We start off not knowing where the robot is.

Example: Robot Localization



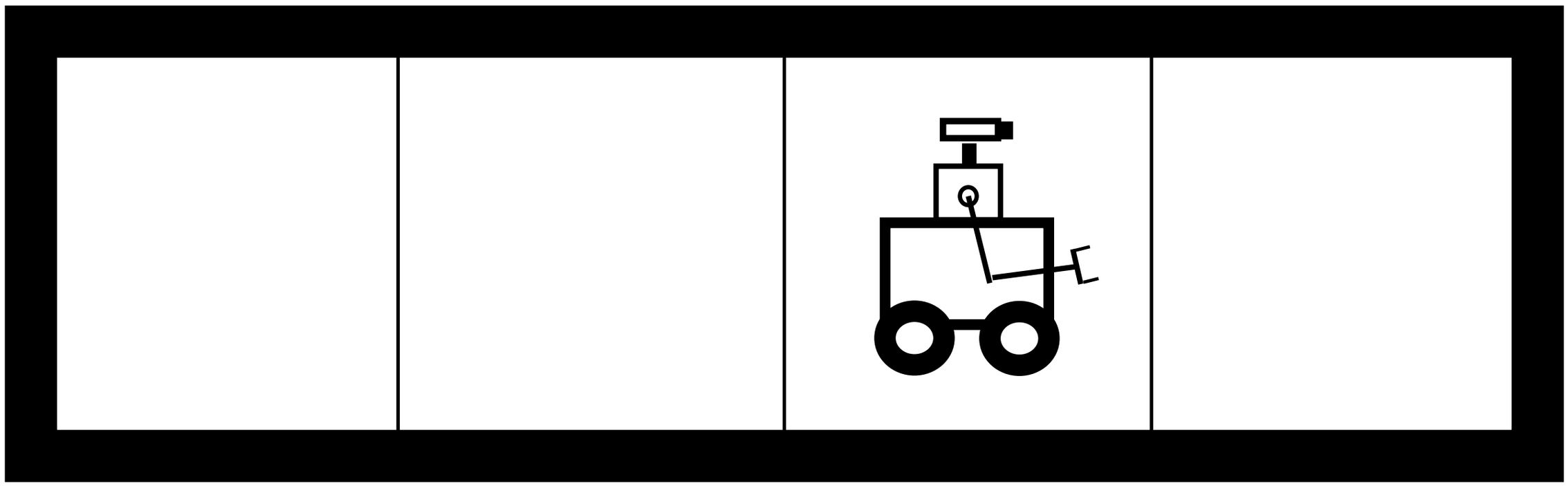
Robot sense: obstacles up and down.
Updates distribution.

Example: Robot Localization



Robot moves right: updates distribution.

Example: Robot Localization



Obstacles up and down, updates distribution.

What Happened

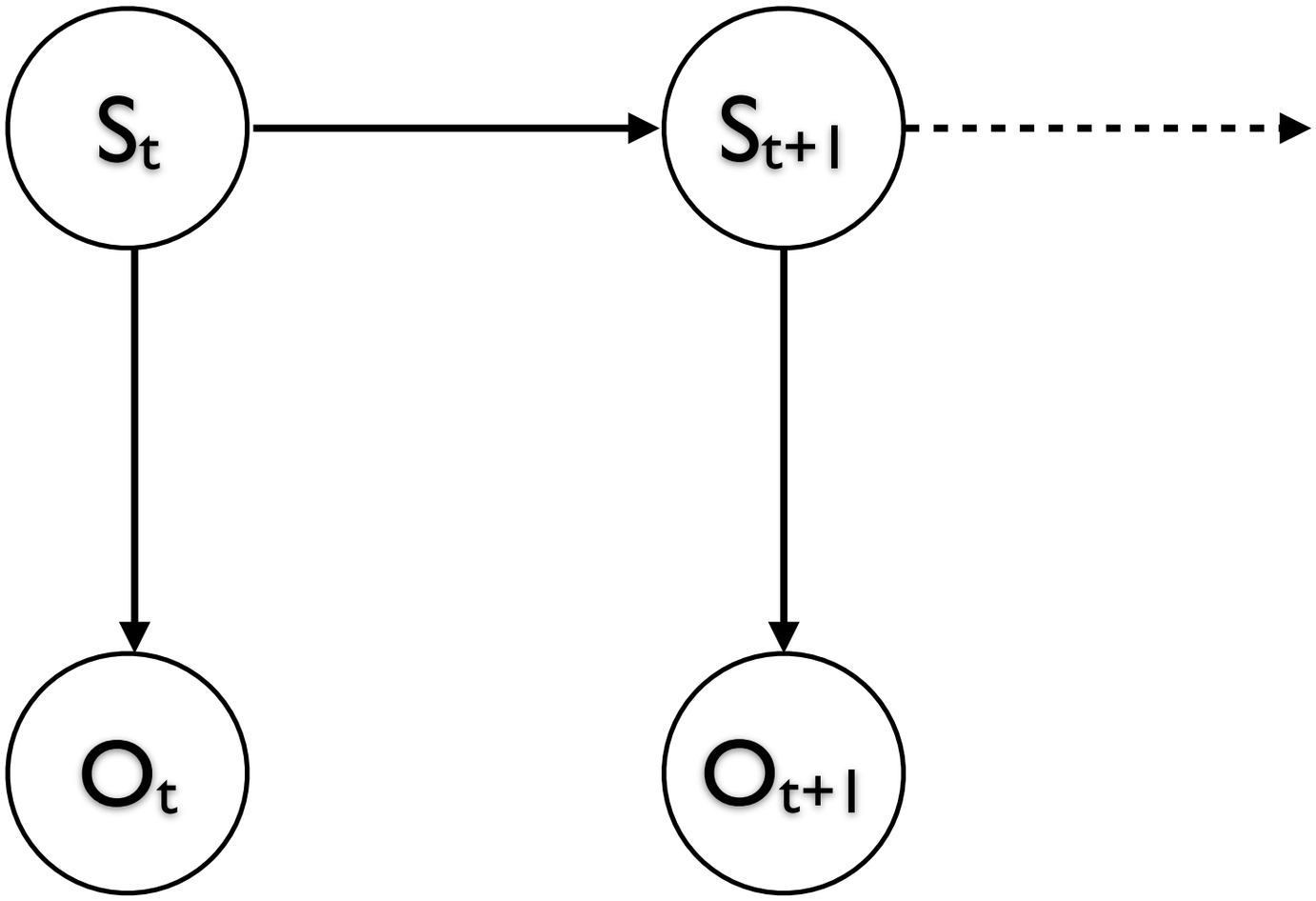
This is an instance of robot tracking.

Could also:

- Predict (where will the robot be in 3 steps?)
- Smooth (where was the robot?)
- Most likely path (what was the robot's path?)

All of these are questions about the HMM's state at various times.

Most Likely Path



How to compute? (assume we have CPTs)

Most Likely State

(Assume all start states equally likely)

Consider a path of length 1.

- $P(S_0) = P(S_0 \mid O_0)$.
- We can just write this down.

Now, what about a path of length 2?

- $P(S_1) = P(S_0)\, P(S_1 \mid S_0)\, P(S_1 \mid 0_1)$
- $= P(S_0 \mid O_0)\, P(S_1 \mid S_0)\, P(S_1 \mid 0_1)$

See the recursion?

Viterbi Algortihm

Most likely path $S_0 \ldots S_n$:

For each state S_k:

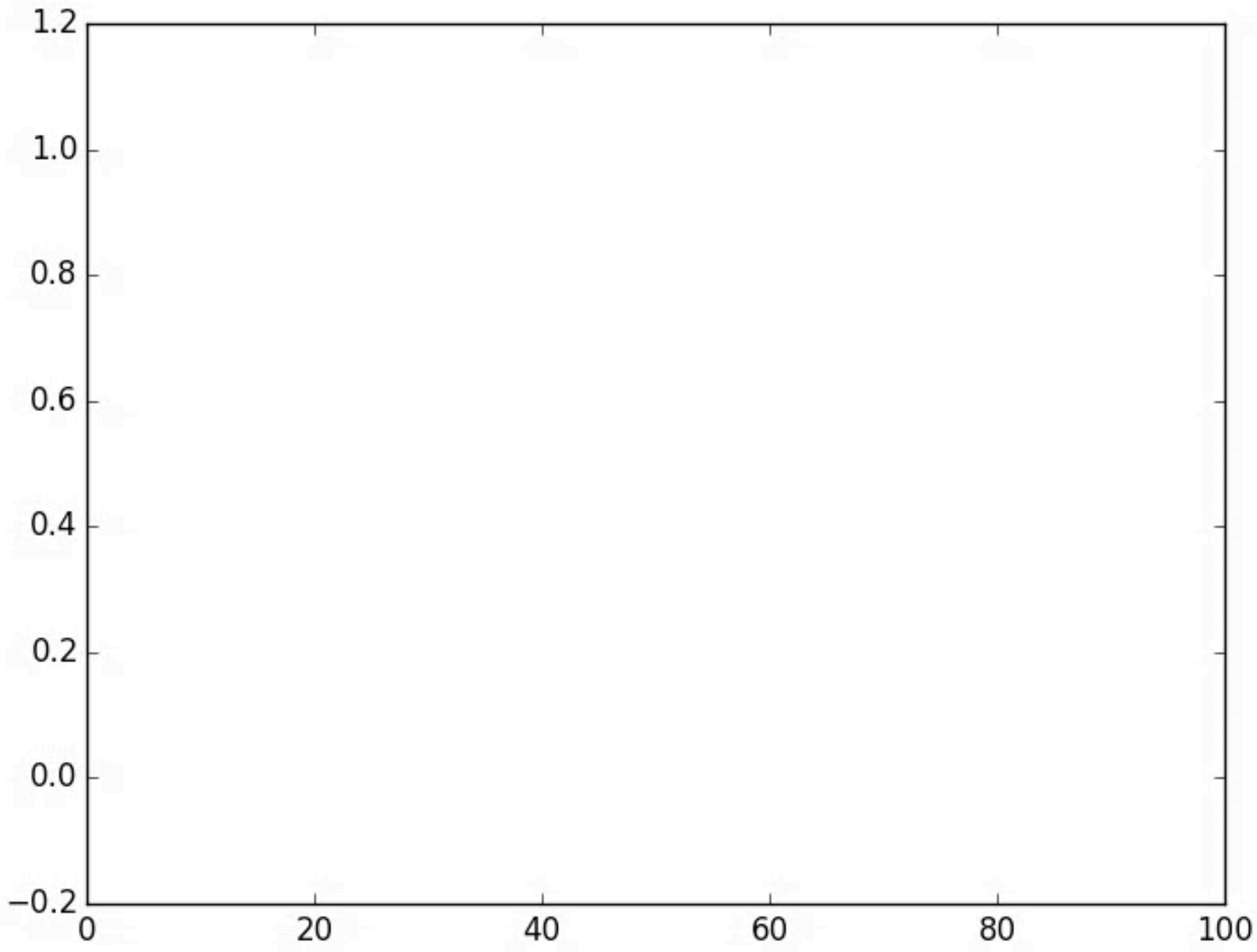
$V_{0,k} = P(O_0 \mid S_k)$

For *i* = *1…n*,

For each *k*:

$V_{i,k} = \max_x P(O_i \mid S_k)\ P(S_k \mid S_x)\ V_{i-1,x}$

Recursive, but values repeat, so save them (dynamic programming).



Viterbi

"The algorithm has found universal application in decoding the convolutional codes used in both CDMA and GSM digital cellular, dial-up modems, satellite, deep-space communications, and 802.11 wireless LANs." (wikipedia)

(photo credit: MIT)