CS440/ECE448 Lecture 16: HMM Inference and the Viterbi Algorithm

Mark Hasegawa-Johnson, 3/2021

Louis-Leopold Boilly, Passer Payez, 1803. Public domain work of art, https://en.wikipedia.org/wiki/Umbrella

Outline

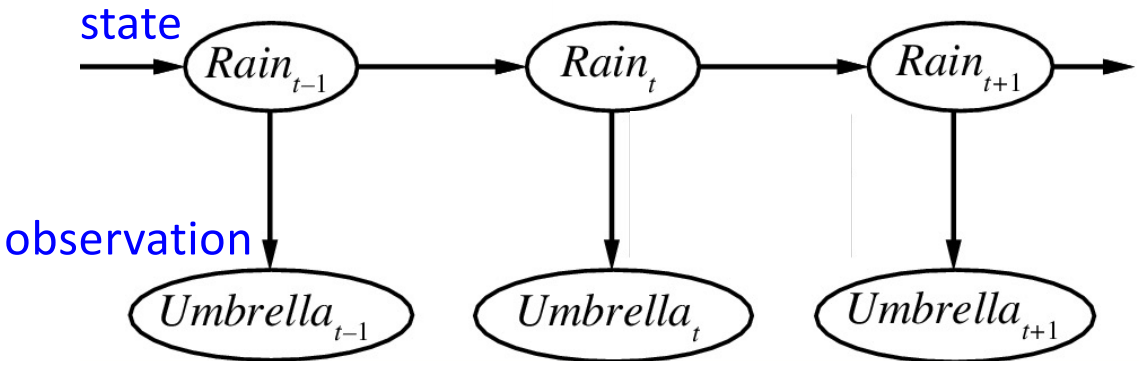
- Inference by Enumeration in an HMM
- Decoding using the Viterbi Algorithm

Example Scenario: UmbrellaWorld

Characters from the novel *Hammered* by Elizabeth Bear,
Scenario from chapter 15 of Russell & Norvig

Since he has read a lot about rain, Richard proposes a hidden Markov model:

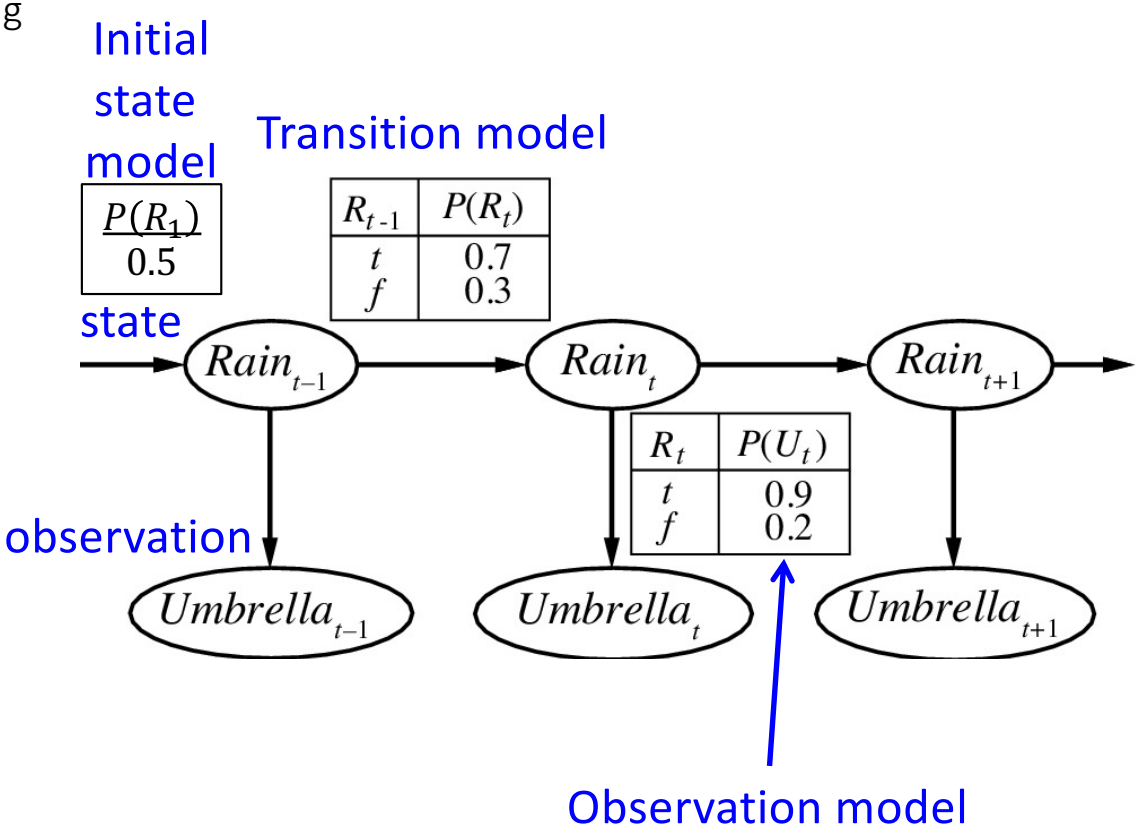
- Rain on day t-1 (R_{t-1}) makes rain on day t (R_t) more likely.
- Elspeth usually brings her umbrella (U_t) on days when it rains (R_t), but not always.



Example Scenario: UmbrellaWorld

Characters from the novel *Hammered* by Elizabeth Bear,
Scenario from chapter 15 of Russell & Norvig

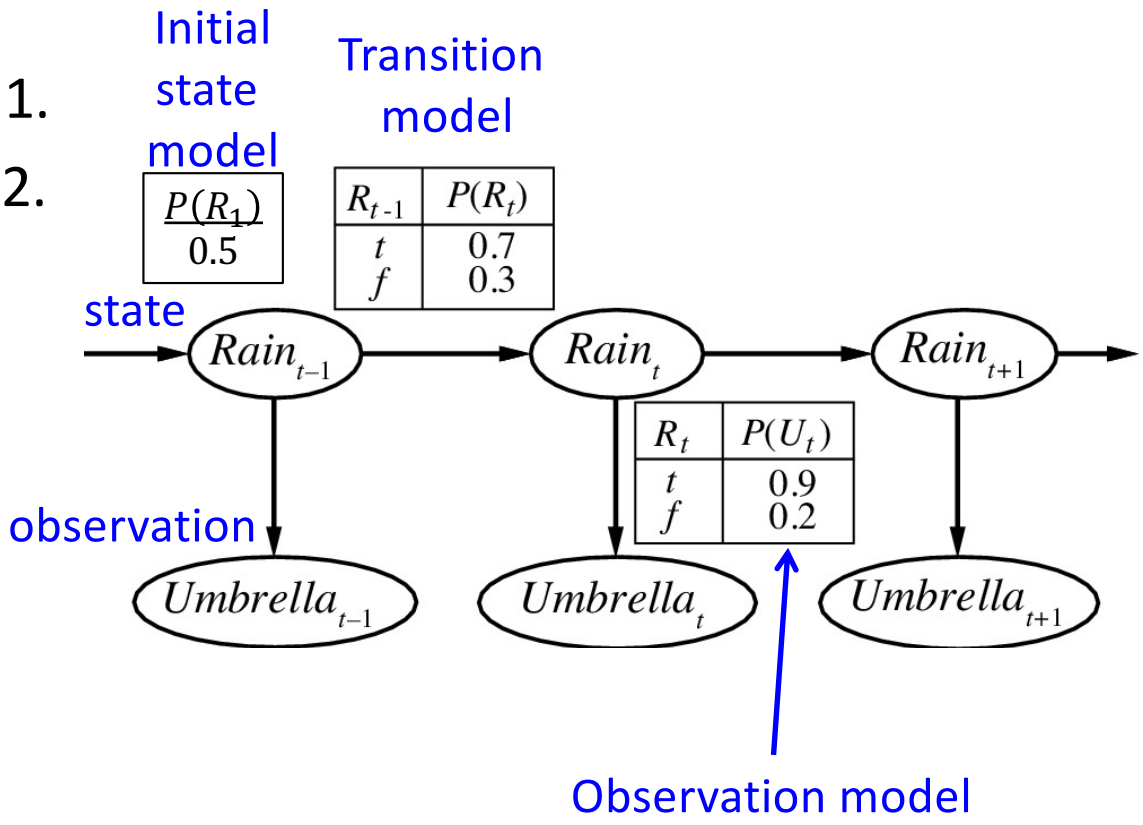
- Richard has no idea whether or not it's raining on day 1, so he assumes $P(R_1) = 0.5$.
- Richard learns that the weather changes on 3 out of 10 days, thus
$$P(R_t|R_{t-1}) = 0.7$$
$$P(R_t|\neg R_{t-1}) = 0.3$$
- He also learns that Elspeth sometimes forgets her umbrella when it's raining, and that she sometimes brings an umbrella when it's not raining. Specifically,
$$P(U_t|R_t) = 0.9$$
$$P(U_t|\neg R_t) = 0.2$$



Inference in an HMM: Example

- Elspeth has no umbrella on day 1.
- Elspeth has an umbrella on day 2.
- Assume $P(R_1) = 0.5$
- What is the probability that it's raining on day 2?

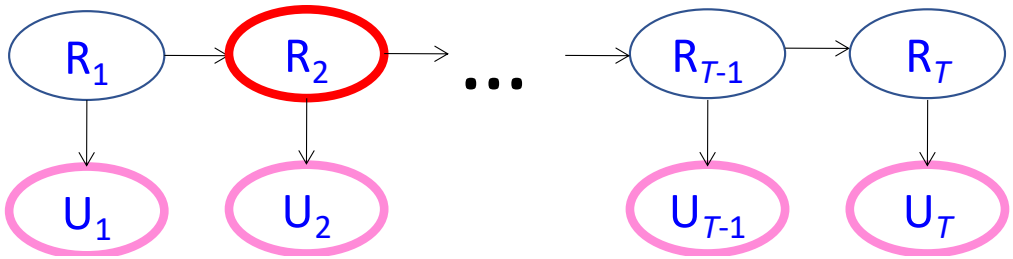
$$P(R_2|\neg U_1, U_2)?$$



Inference by Enumeration

To calculate a probability $P(R_2|U_1,U_2)$:

1. **Select:** which variables do we need, in order to model the relationship among U_1, U_2, and R_2?
   - We need also R_1.
2. **Multiply** to compute joint probability:
   $$P(R_1, R_2, U_1,U_2) = P(R_1)P(U_1|R_1) \dots P(U_2|R_2)$$
3. **Add** to eliminate those we don't care about
   $$P(R_2, U_1,U_2) = \sum_{R_1} P(R_1, R_2, U_1,U_2)$$
4. **Divide:** use Bayes' rule to get the desired conditional
   $$P(R_2|U_1,U_2) = P(R_2, U_1,U_2)/P(U_1,U_2)$$



Computational Complexity of "Inference by Enumeration"

- Russell & Norvig call this "inference by enumeration" because you have to enumerate every possible combination of R_1, R_2, U_1, U_2.
- The complexity comes from this enumeration: if there are 2^4 possible combinations, then the complexity can't be less than 2^4!

First simplification for HMMs: only enumerate the values of the hidden variables

- Notice: we don't really need to calculate $P(\neg R_1, R_2, \neg U_1, \neg U_2)$ if we have already observed that U_2 is True!
- First computational simplification for HMMs:
  - Only enumerate the possible values of the hidden variables.
  - Set the observed variables to their observed values.

Inference by Enumerating only the Hidden Variables

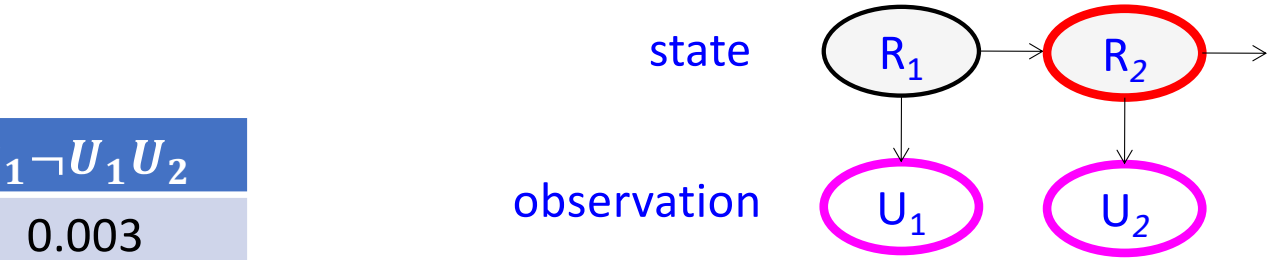
Multiply:

$$P(R_1, R_2, \neg U_1, U_2) = P(R_1)P(\neg U_1|R_1)P(R_2|R_1)P(U_2|R_2)$$

	$\neg R_1 \neg U_1 U_2$	$R_1 \neg U_1 U_2$
$\neg R_2$	0.056	0.003
R_2	0.108	0.0315

- We only compute joint probabilities that include the observed events, $\neg U_1$ and U_2.
- The numbers don't add up to one; they add up to $P(\neg U_1, U_2)$.

Transition model



Transition probabilities

	R_t = T	R_t = F
R_{t-1} = T	0.7	0.3
R_{t-1} = F	0.3	0.7

Observation probabilities

	U_t = T	U_t = F
R_t = T	0.9	0.1
R_t = F	0.2	0.8

Inference by Enumerating only the Hidden Variables

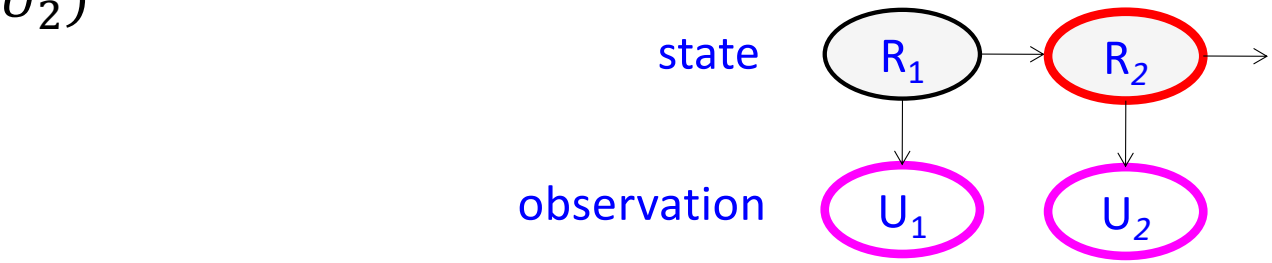
Add:

$$P(R_2, \neg U_1, U_2) = \sum_{R_1} P(R_1, R_2, \neg U_1, U_2)$$

	$\neg U_1 U_2$
$\neg R_2$	0.059
R_2	0.1395

- We only compute joint probabilities that include the observed events, $\neg U_1$ and U_2.
- The numbers don't add up to one; they add up to $P(\neg U_1, U_2)$.

Transition model



Transition probabilities

	R_t = T	R_t = F
R_{t-1} = T	0.7	0.3
R_{t-1} = F	0.3	0.7

Observation probabilities

	U_t = T	U_t = F
R_t = T	0.9	0.1
R_t = F	0.2	0.8

Inference by Enumerating only the Hidden Variables

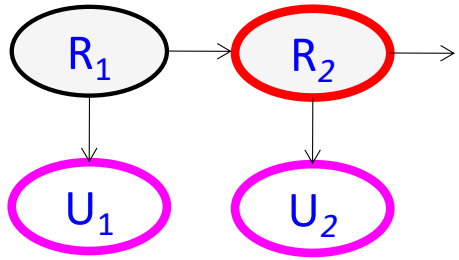
Divide:

$$P(R_2|\neg U_1,U_2) = \frac{P(R_2,\neg U_1,U_2)}{P(\neg U_1,U_2)}$$

	$\neg U_1 U_2$
$\neg R_2$	0.30
R_2	0.70

- Normalize, so that the column sums to one.

Transition model



Transition probabilities

	R_t = T	R_t = F
R_{t-1} = T	0.7	0.3
R_{t-1} = F	0.3	0.7

Observation probabilities

	U_t = T	U_t = F
R_t = T	0.9	0.1
R_t = F	0.2	0.8

First simplification for HMMs: only enumerate the values of the hidden variables

- Only enumerate the possible values of the hidden variables. Set the observed variables to their observed values.
- **Inference with binary hidden variables**: enumerate $(R_1, \dots, R_T)$, complexity is $2^{T+1} = \mathcal{O}\{2^T\}$.
- **Inference with N-ary hidden variables:** If each of the variables R_t has N possible values, instead of only 2 possible values, then the inference complexity would be $\mathcal{O}\{N^T\}$.

Outline

- Inference by Enumeration in an HMM
- Decoding using the Viterbi Algorithm

Inference complexity in an HMM

- $\mathcal{O}\{N^T\}$ is still a lot. Can we do better?
- For a general Bayes net, no. Bayes net inference, in an arbitrary Bayes net, is NP-complete.
- For an HMM, yes, we can do better.

Hard EM for an HMM: the Viterbi Algorithm

The Viterbi algorithm is the inference step of hard EM for an HMM.

Given a particular sequence of observations, what is the most likely underlying sequence of states?

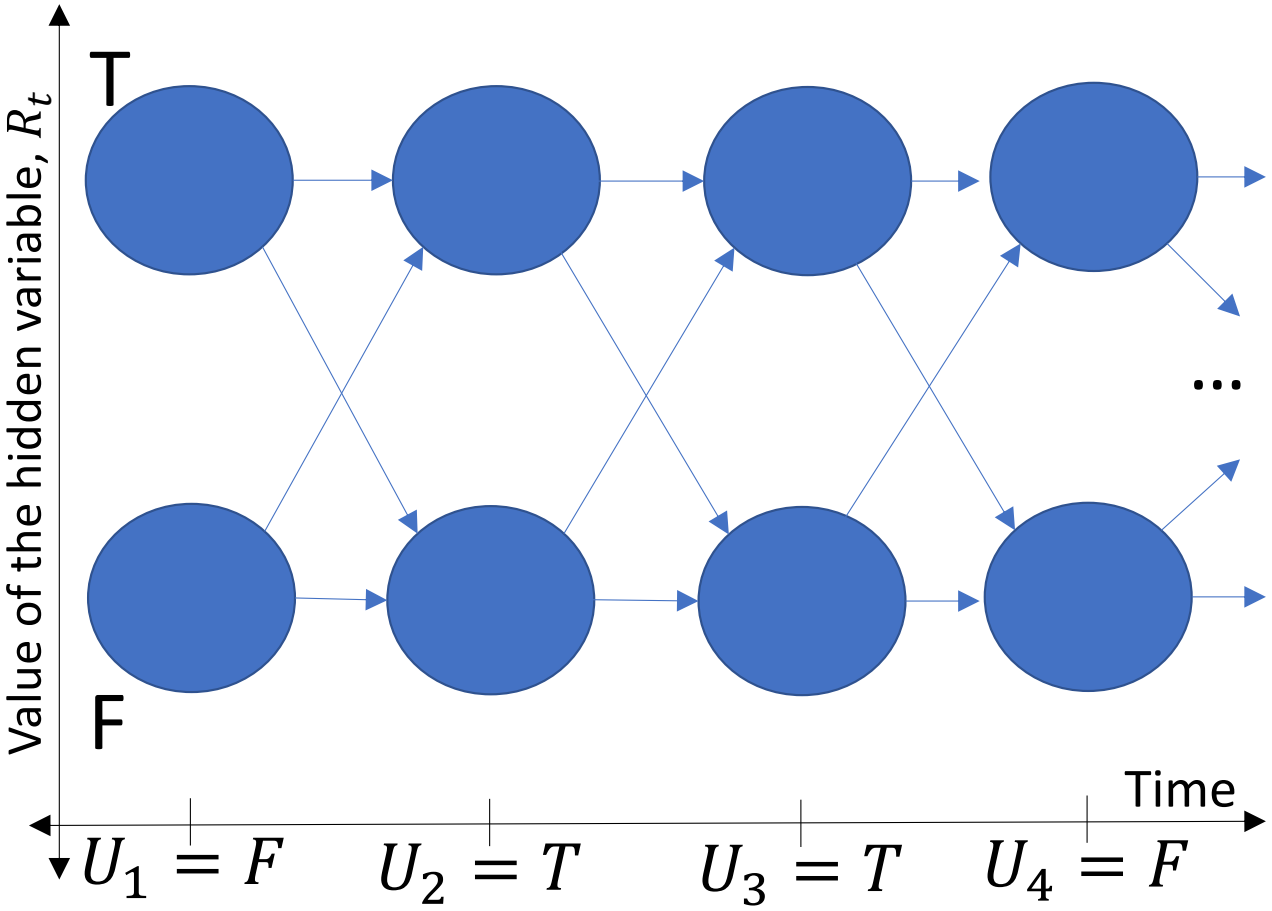
Viterbi Algorithm Example

Given a particular sequence of observations, what is the most likely underlying sequence of states?

- Example: given $U_1 = F, U_2 = T, U_3 = T, U_4 = F$
- what is the most likely sequence of state variables, R_1, R_2, R_3, R_4?

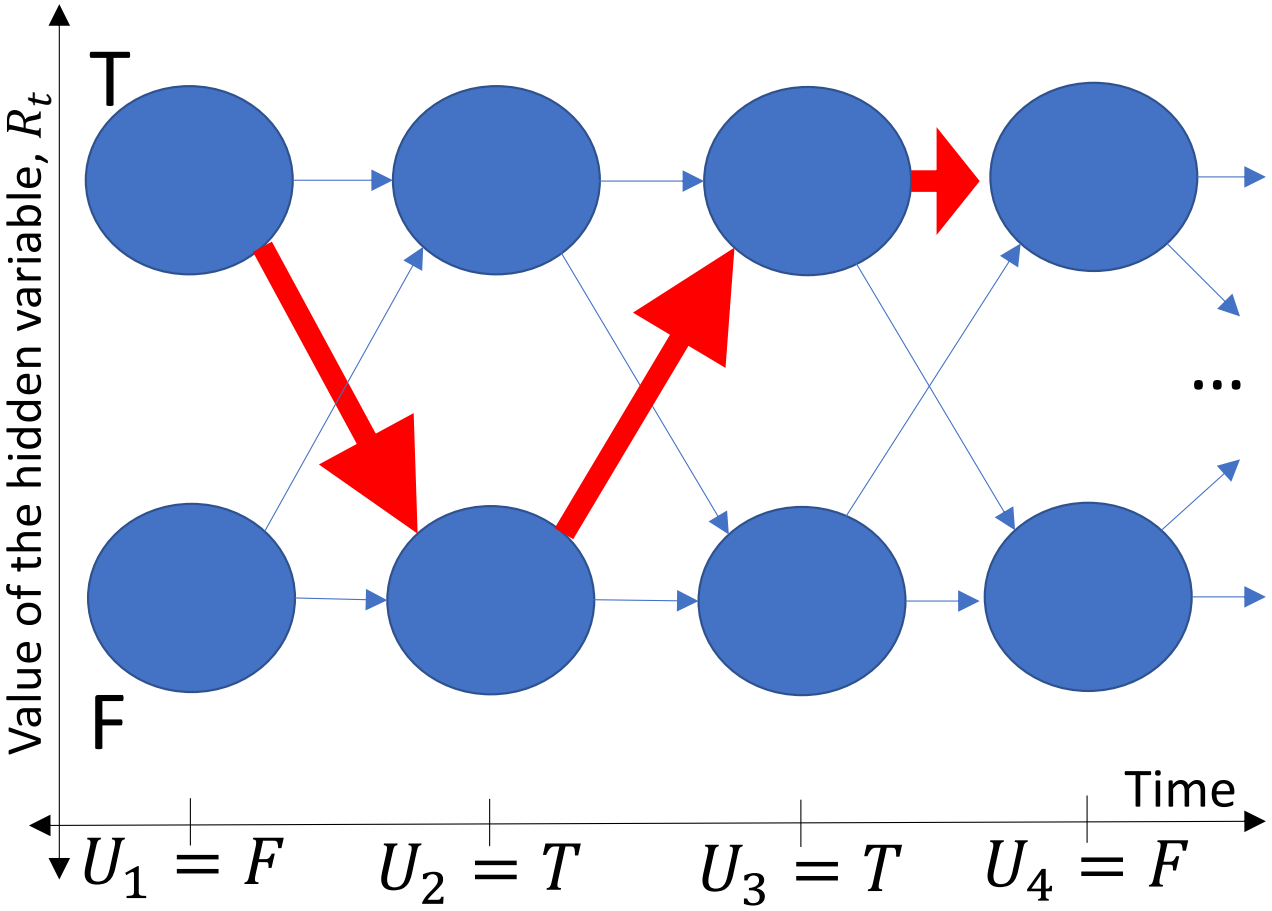
The Trellis

- X-Axis = time
- Y-Axis = state variable (rt)
- Node = a particular state at a particular time
- Edge = possible transition from R_{t-1} to R_t



A Path Through the Trellis

- A path through the trellis is a sequence of connected states.
- For example, this path is the sequence $R_1 = T, R_2 = F, R_3 = T, R_4 = T$



Viterbi Algorithm Key Concept

Given a particular sequence of observations, what is the most likely underlying sequence of states?

In other words, given a particular sequence of observations, what is the most probable path through the trellis?

Viterbi Algorithm: Key concepts

Nodes and edges have numbers attached to them:

- **Edge Probability**: Probability of taking that transition, and then generating the next observed output

$$e_{ijt} = P(R_t = j, U_t = u_t | R_{t-1} = i)$$

- **Node Probability**: Probability of the best path until node j at time t

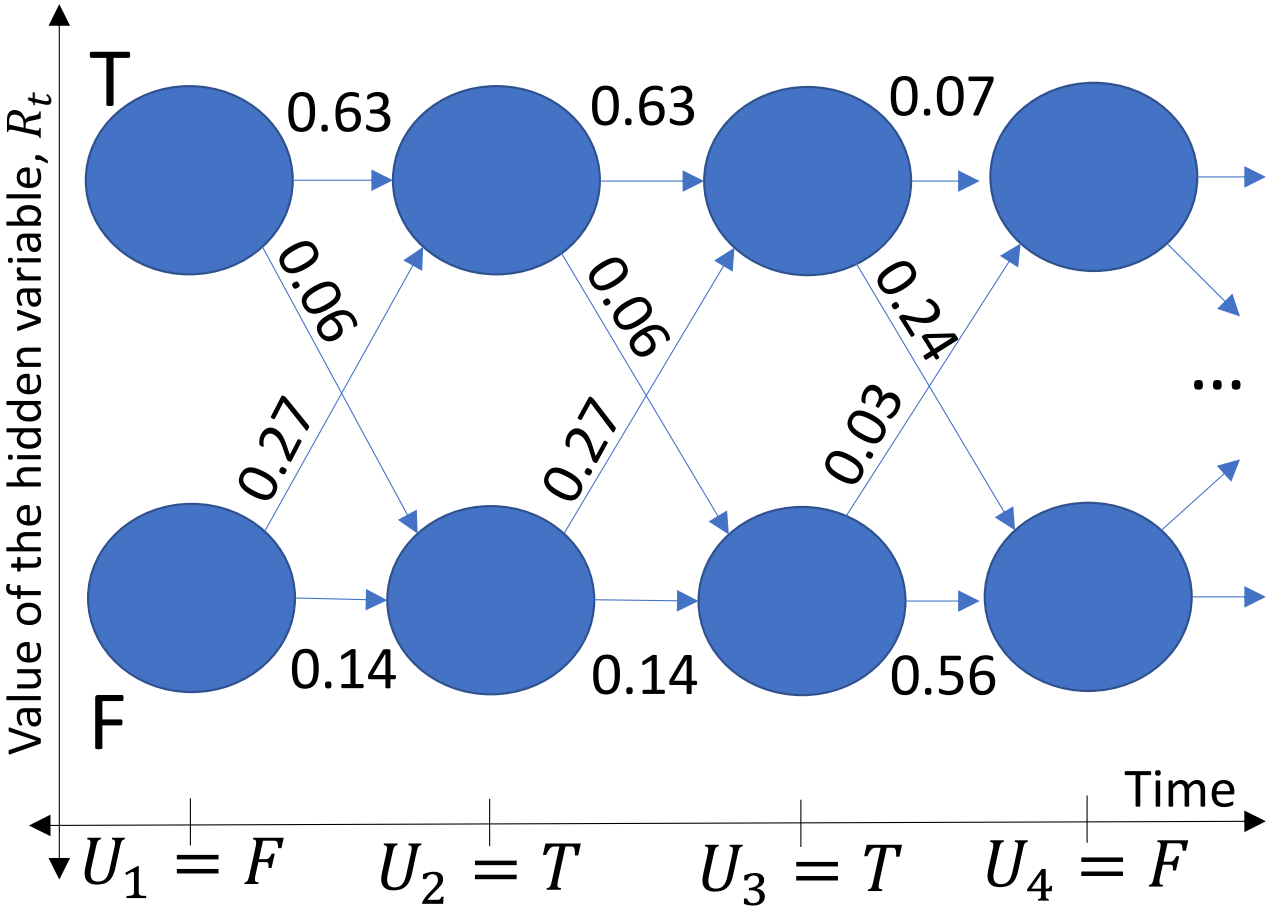
$$v_{jt} = \max_{r_1,\ldots,r_{t-1}} P(U_1 = u_1 \ldots, U_t = u_t, R_1 = r_1, \ldots, R_t = j)$$

Edge Probabilities

$$e_{ijt} = P(R_t = j, U_t = u_t | R_{t-1} = i)$$

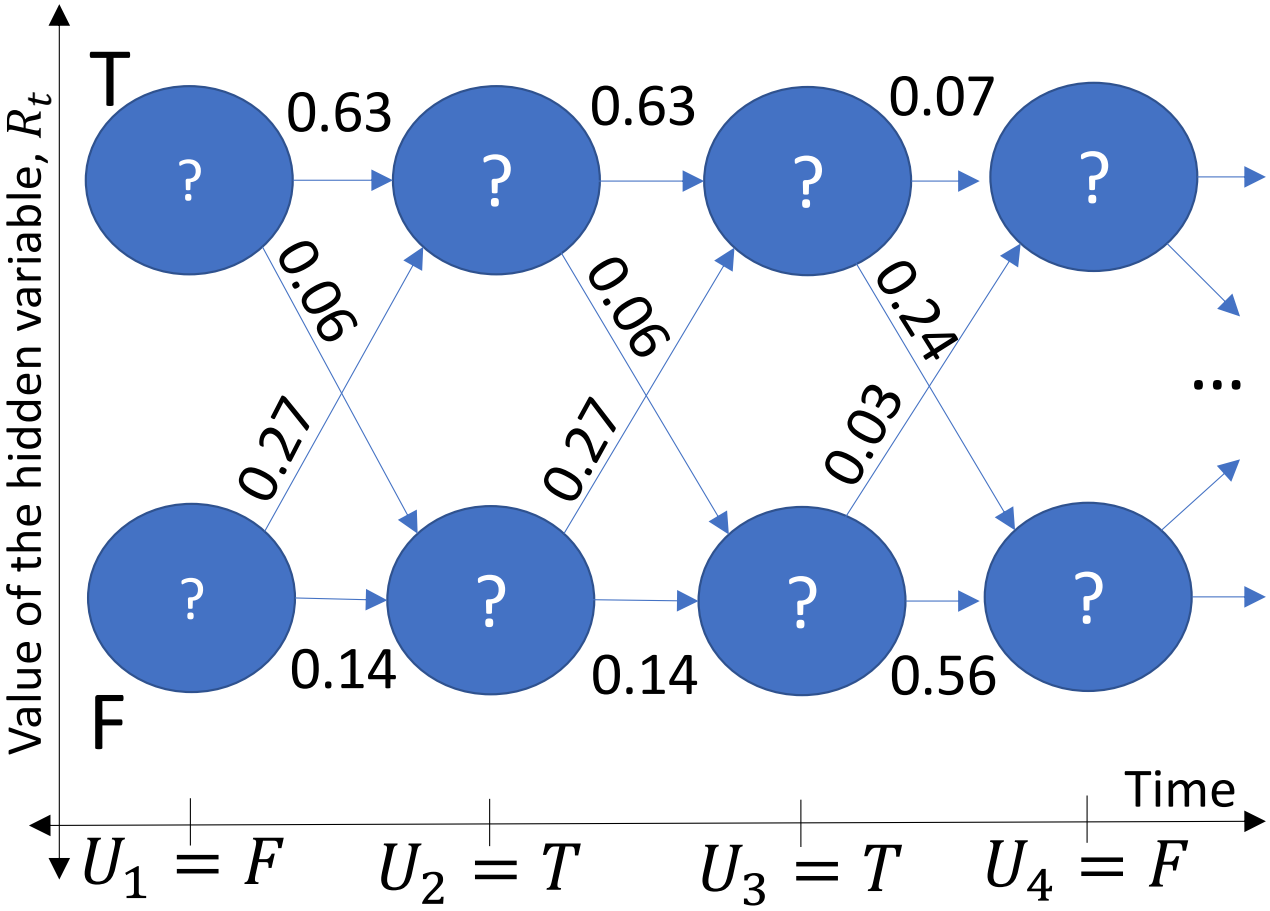
$$= P(R_t = j | R_{t-1} = i) \times P(U_t = u_t | R_t = j)$$

Notice that, since U_2 and U_3 have the same observed values, their inbound edges have the same weights.



Node Probabilities

$$v_{jt} = \max_{r_1,\ldots,r_{t-1}} P(U_1 = u_1 \ldots, U_t = u_t, R_1 = r_1, \ldots, R_t = j)$$

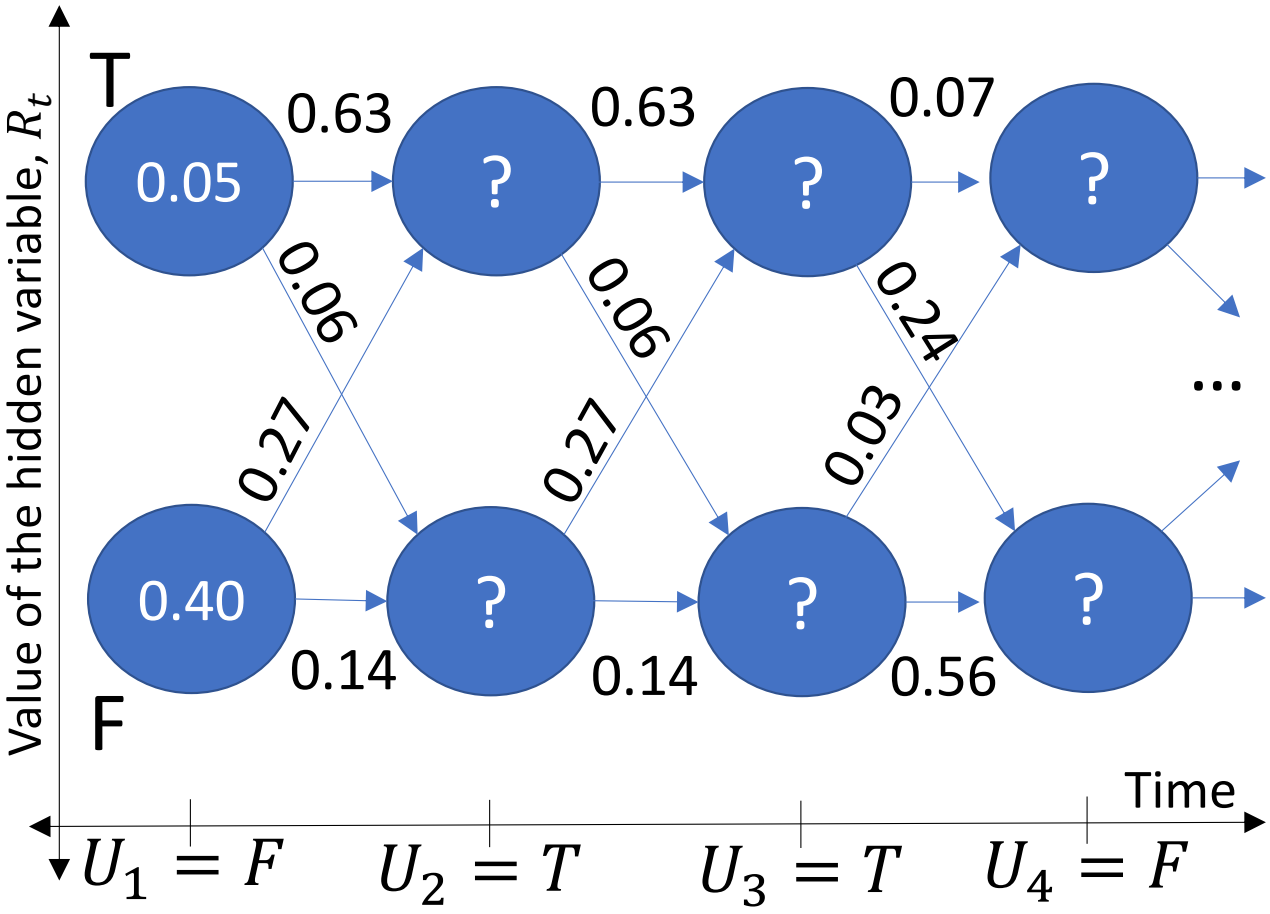


Node Probabilities: Initialization

For example, let's consider how to find v_{jt} for $t = 1$:

$$v_{j1} = P(U_1 = u_1, R_1 = j)$$
$$= P(R_1 = j)$$
$$\times P(U_1 = u_1 | R_1 = j)$$

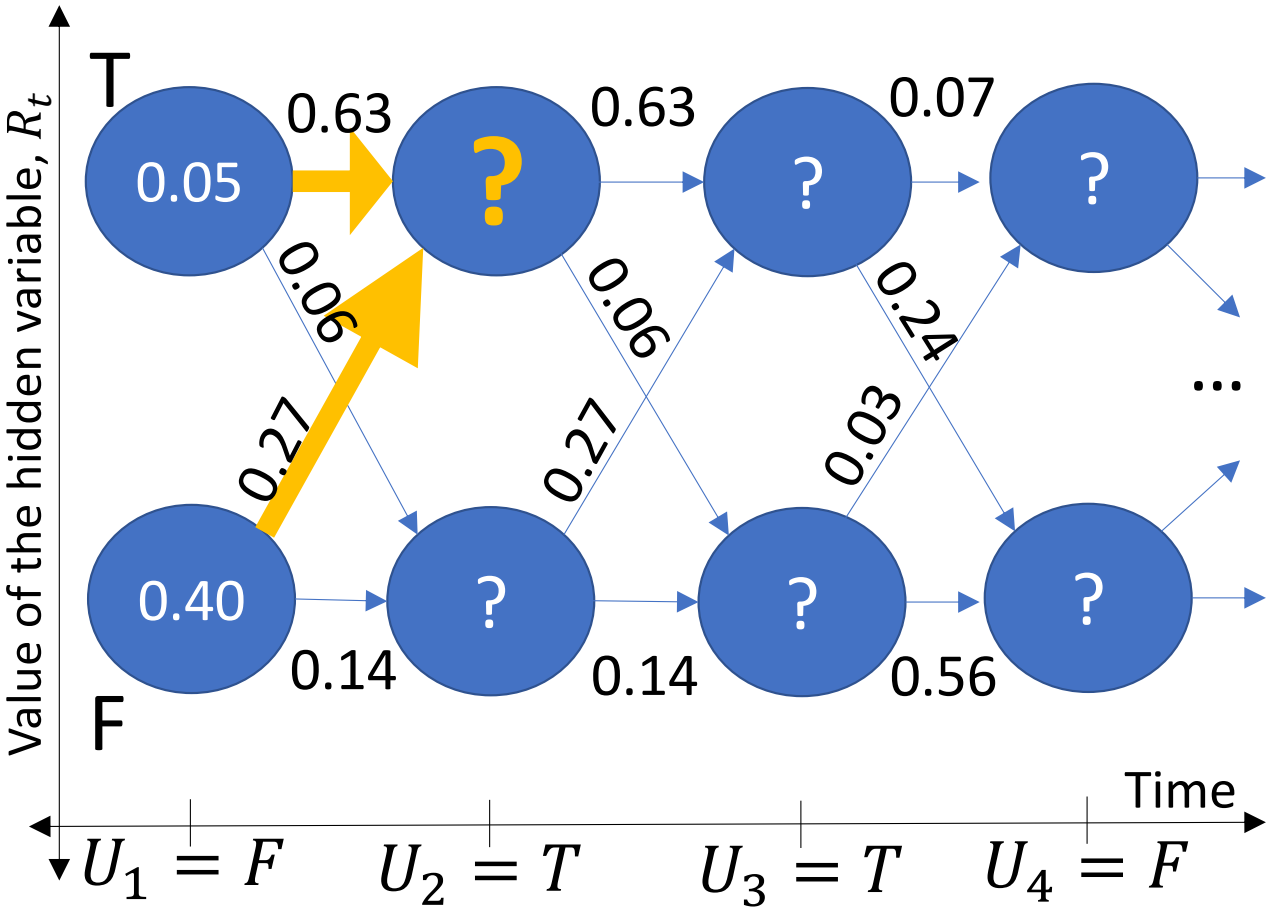
$$= \begin{cases} (0.5)(0.1) & j = T \\ (0.5)(0.8) & j = F \end{cases}$$



… and what about time t=2?

Notice that, at time t=2, there are two ways to get to any particular state:

- The previous state might have been F
- The previous state might have been T



Viterbi Algorithm: the iteration step

Given edge probabilities defined as

$$e_{ijt} = P(R_t = j, U_t = u_t | R_{t-1} = i)$$

and node probabilities defined as

$$v_{jt} = \max_{r_1,\dots,r_{t-2},i} P(U_1 = u_1 \dots, U_t = u_t, R_1 = r_1, \dots, R_{t-1} = i, R_t = j)$$

The node probability can be efficiently computed as

$$\begin{aligned} & v_{jt} \\ & = \max_i \left(\max_{r_1,\dots,r_{t-2}} P(U_1 = u_1 \dots, U_{t-1} = u_{t-1}, R_1 = r_1, \dots, R_{t-1} = i) \right. \\ & \left. \times P(R_t = j, U_t = u_t | R_{t-1} = i) \right) \end{aligned}$$

Viterbi Algorithm: the iteration step

Given edge probabilities defined as

$$e_{ijt} = P(R_t = j, U_t = u_t | R_{t-1} = i)$$

and node probabilities defined as

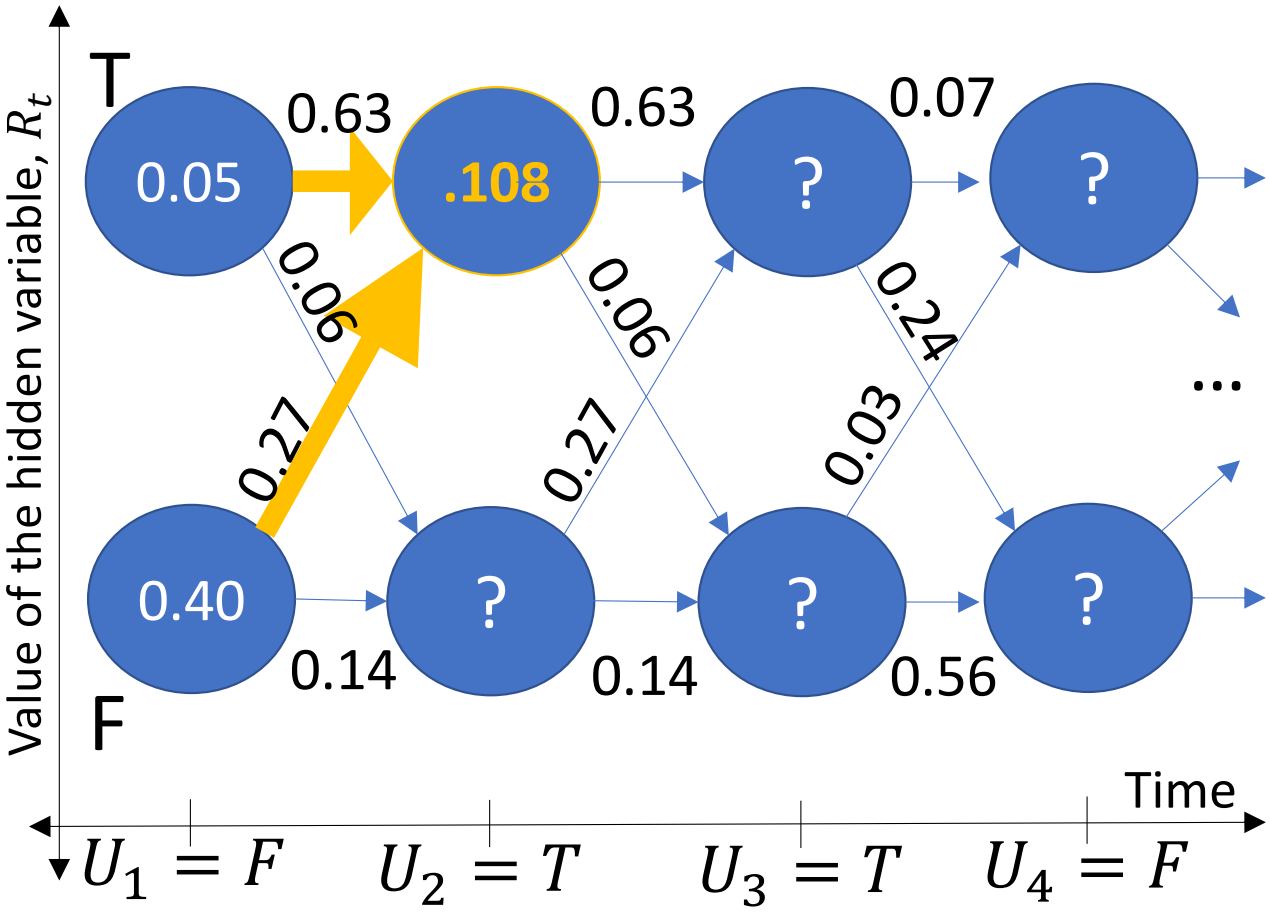
$$v_{jt} = \max_{r_1,\ldots,r_{t-2},i} P(U_1 = u_1 \ldots, U_t = u_t, R_1 = r_1, \ldots, R_{t-1} = i, R_t = j)$$

The node probability can be efficiently computed as

$$v_{jt} = \max_i v_{i,t-1} e_{ijt}$$

… and what about time t=2?

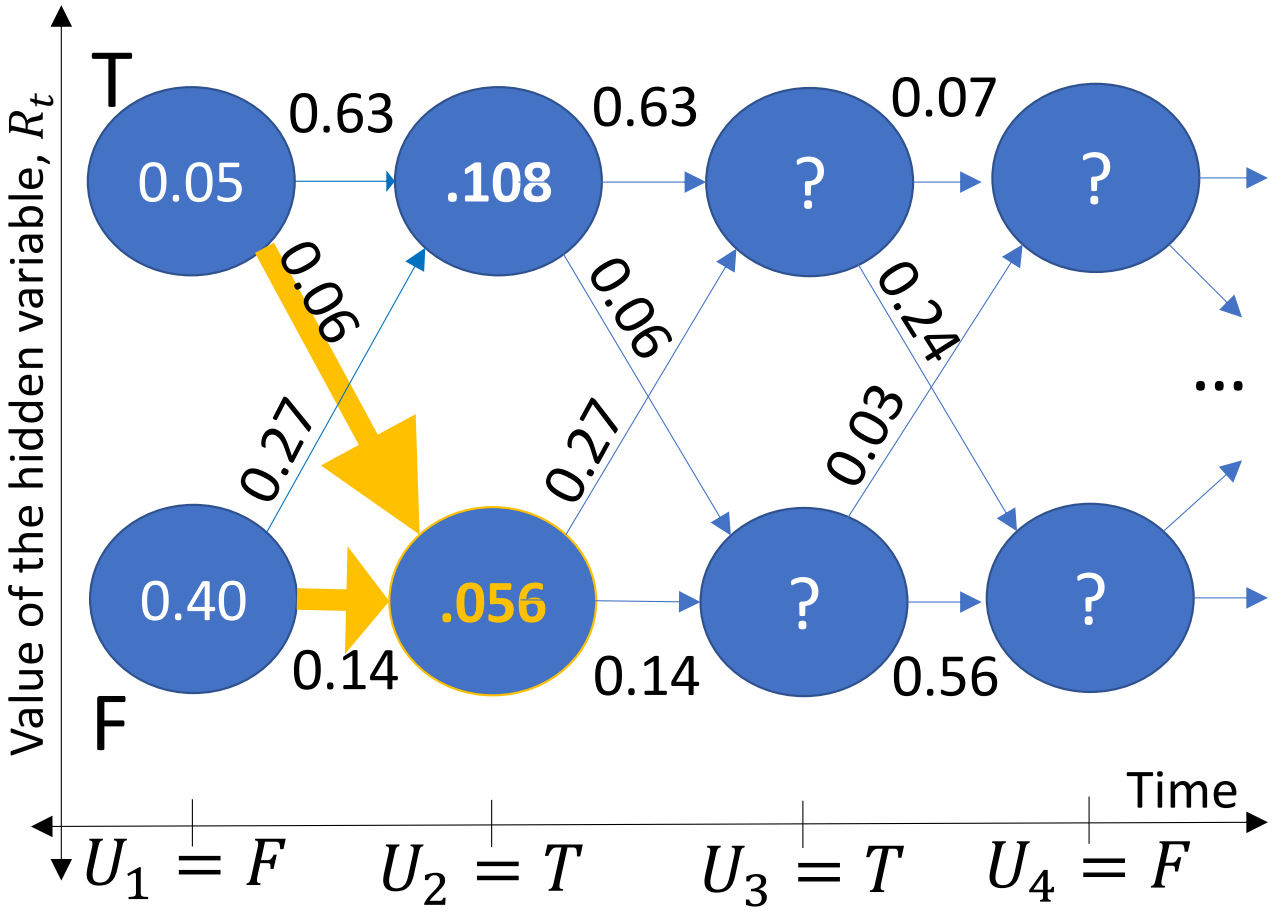
$$v_{T,2} = \max_i \; v_{i1} e_{i,T,2}$$
$$= \max\big((0.05)(0.63), (0.4)(0.27)\big)$$
$$= \; 0.108$$



… and what about time t=2?

$$v_{T,2} = \max_i v_{i1} e_{i,T,2}$$
$$= \max\big((0.05)(0.63), (0.4)(0.27)\big)$$
$$= 0.108$$

$$v_{F,2} = \max_i v_{i1} e_{i,F,2}$$
$$= \max\big((0.05)(0.06), (0.4)(0.14)\big)$$
$$= 0.056$$



Node probabilities and backpointers

- **Node Probability**: Probability of the best path until node j at time t

$$v_{jt} = \max_{r_1,\ldots,r_{t-2},i} P(U_1 = u_1 \ldots, U_t = u_t, R_1 = r_1, \ldots, R_{t-1} = i, R_t = j)$$

- **Backpointer**: which node, i, precedes node j on the best path?

$$i^*_{jt} = \operatorname*{argmax}_{r_1,\ldots,r_{t-2},i} P(U_1, = u_1 \ldots, U_t = u_t, R_1 = r_1, \ldots, R_{t-1} = i, R_t = j)$$

Backpointers at t=2

$$i^*_{T,2} = \underset{i}{\text{argmax}}\, v_{i1} e_{i,T,2}$$

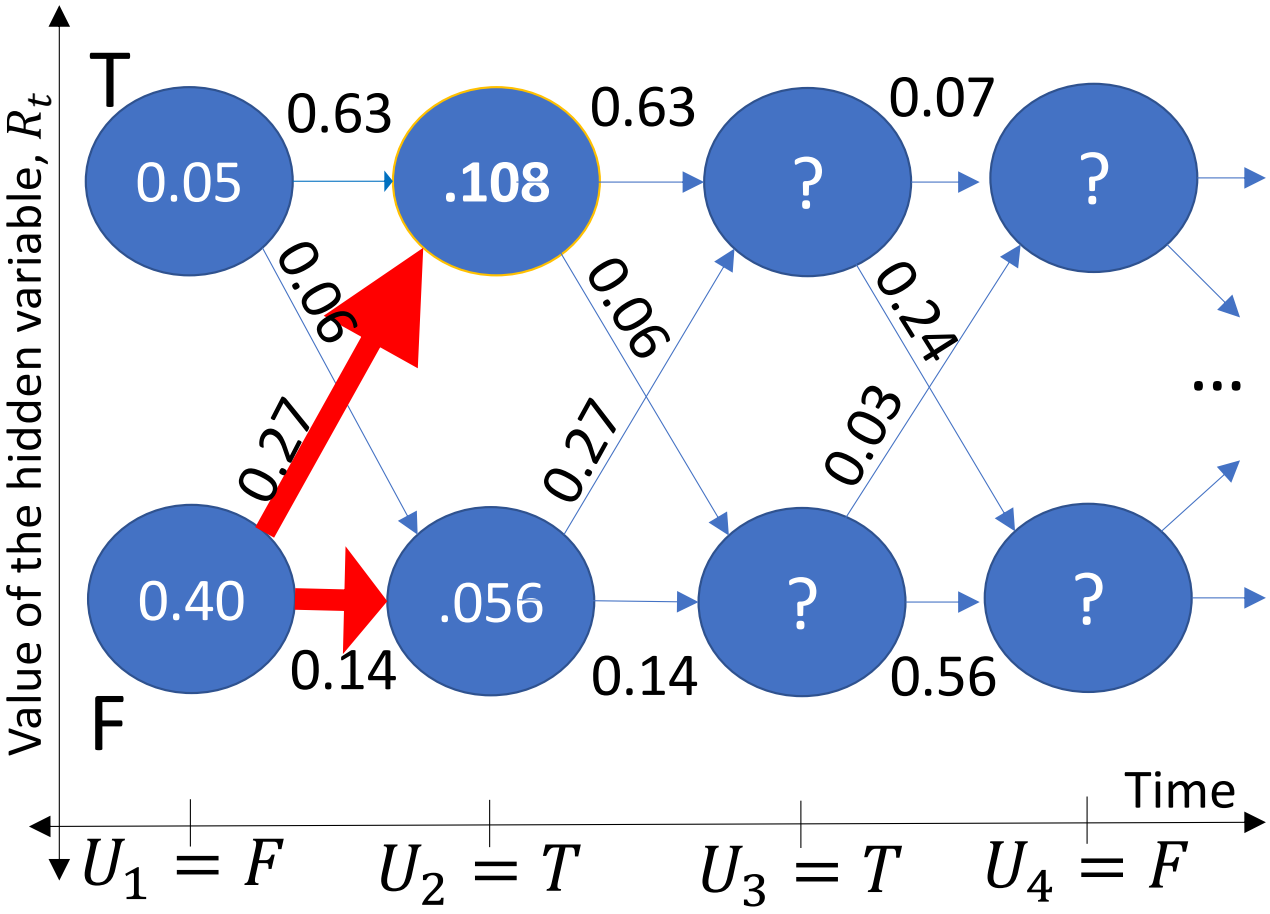
$$= \text{argmax}\begin{pmatrix}(0.05)(0.63),\\ (0.4)(0.27)\end{pmatrix}$$

$$= F$$

$$i^*_{F,2} = \underset{i}{\text{argmax}}\; v_{i1} e_{i,F,2}$$

$$= \text{argmax}\begin{pmatrix}(0.05)(0.06),\\ (0.4)(0.14)\end{pmatrix}$$

$$= F$$



Backpointers at t=3

$$i^*_{T,3} = \underset{i}{\text{argmax}}\, v_{i2} e_{i,T,3}$$

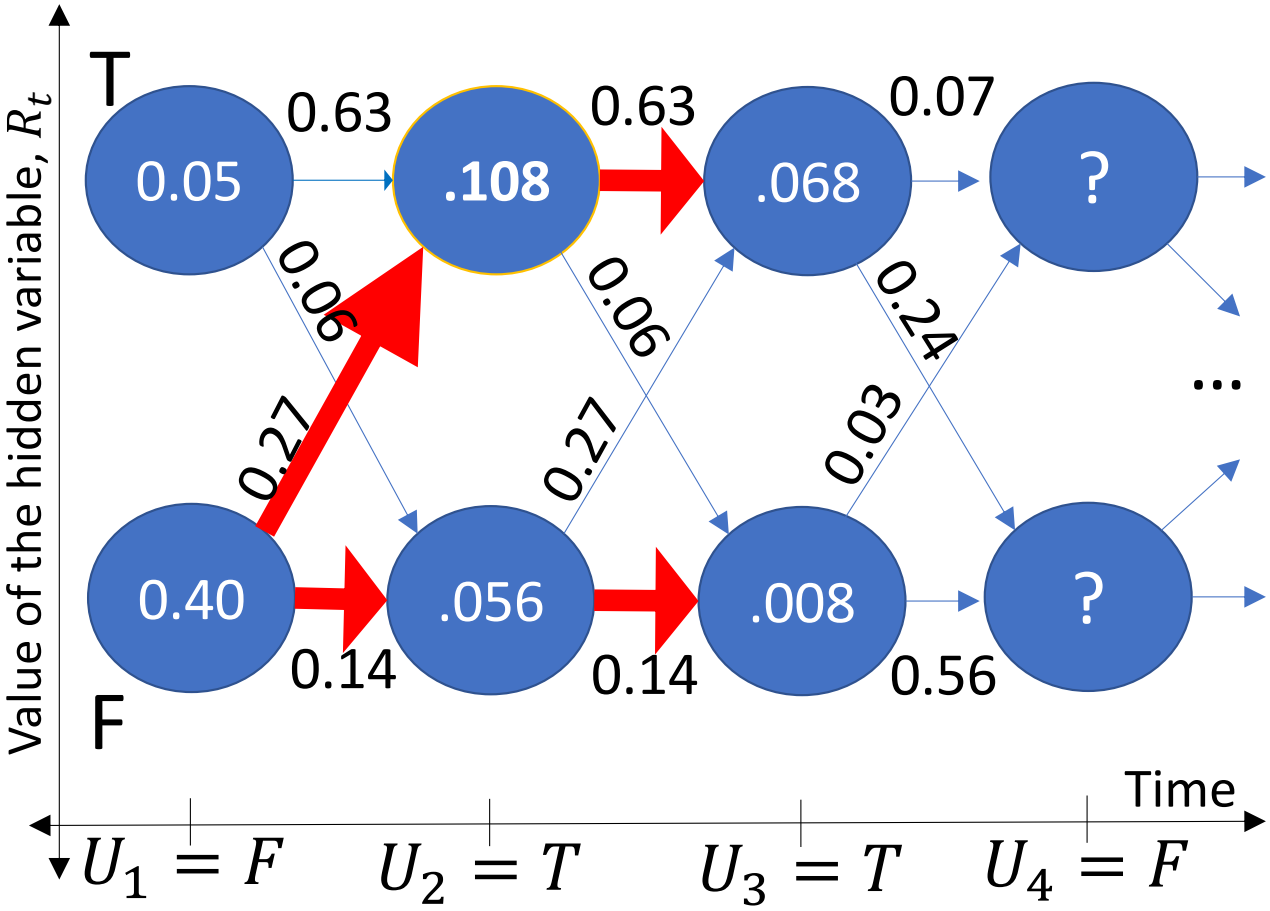
$$= \text{argmax}\begin{pmatrix}(0.108)(0.63),\\(0.056)(0.27)\end{pmatrix}$$

$$= T$$

$$i^*_{F,3} = \underset{i}{\text{argmax}}\, v_{i2} e_{i,F,3}$$

$$= \text{argmax}\begin{pmatrix}(0.108)(0.06),\\(0.056)(0.14)\end{pmatrix}$$

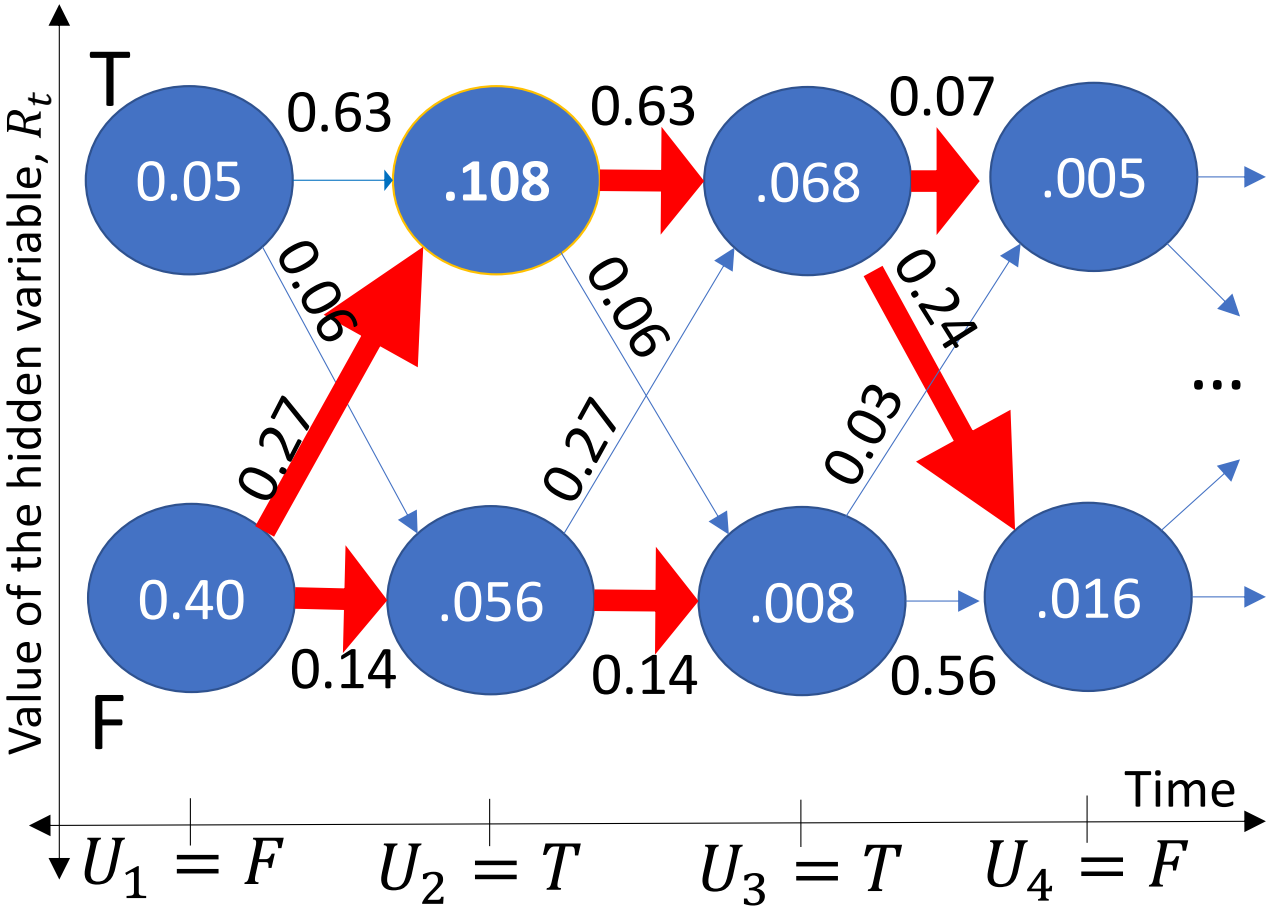
$$= F$$



Backpointers at t=4

$$i^*_{T,4} = \underset{i}{\text{argmax}}\, v_{i3} e_{i,T,4}$$
$$= \text{argmax}\begin{pmatrix}(0.068)(0.07),\\(0.008)(0.03)\end{pmatrix}$$
$$= T$$

$$i^*_{F,4} = \underset{i}{\text{argmax}}\, v_{i3} e_{i,F,4}$$
$$= \text{argmax}\begin{pmatrix}(0.068)(0.24),\\(0.008)(0.56)\end{pmatrix}$$
$$= T$$



So which is the best path?

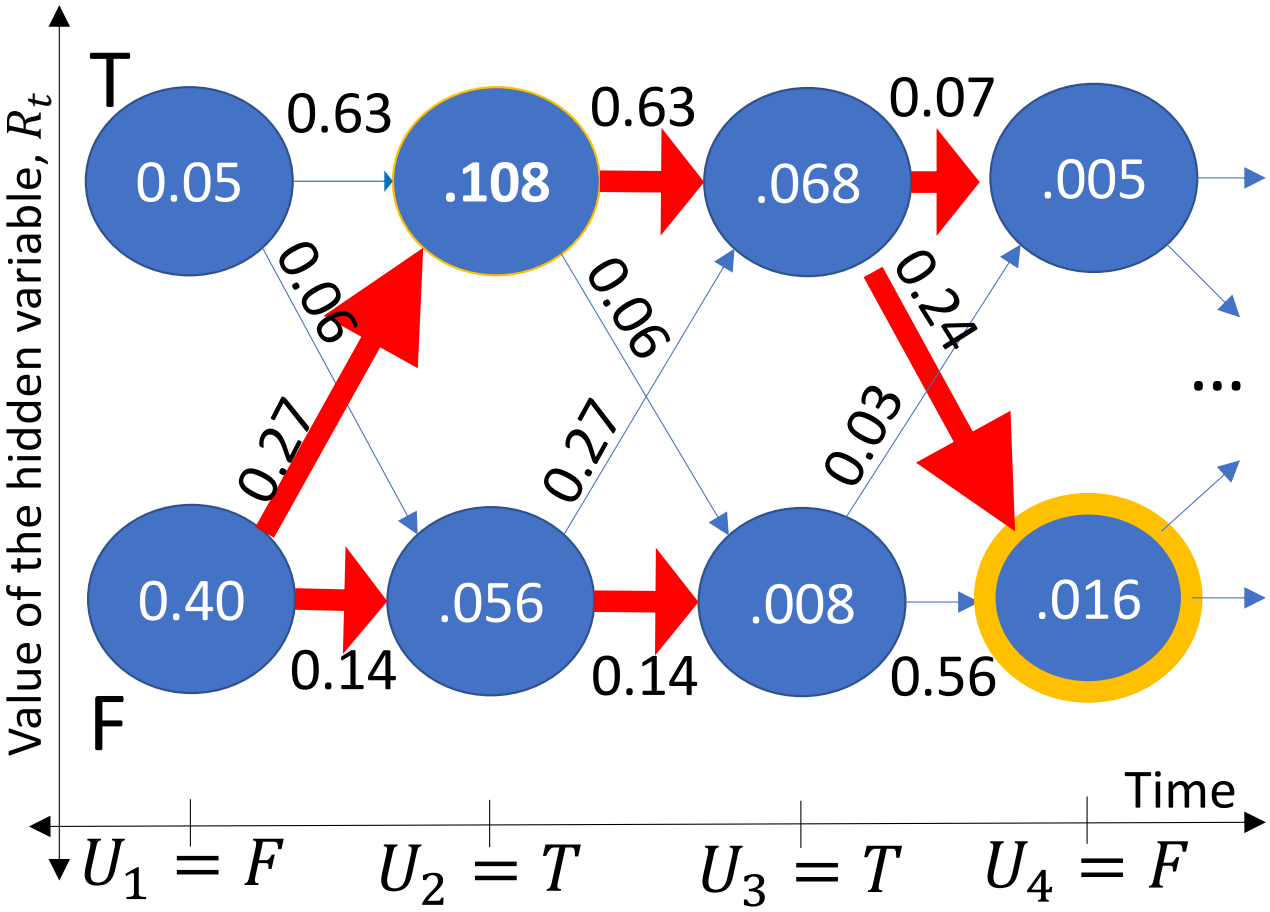
- Answer: whichever one is most probable.

Node probabilities at t=4

$$v_{T,4} = \max_i v_{i3} e_{i,T,4}$$
$$= \max \begin{pmatrix} (0.068)(0.07), \\ (0.008)(0.03) \end{pmatrix}$$
$$= 0.005$$

$$v_{F,4} = \max_i v_{i3} e_{i,F,4}$$
$$= \max \begin{pmatrix} (0.068)(0.24), \\ (0.008)(0.56) \end{pmatrix}$$
$$= 0.016$$

The best path is the one that ends at $R_4 = F$



Termination: which is the best path?

- Best final state is whichever final state has the highest node probability.
- The best path leading to that state is the most probable one
- … but we've already found the most probable path…
- …we just need to follow the backpointers!

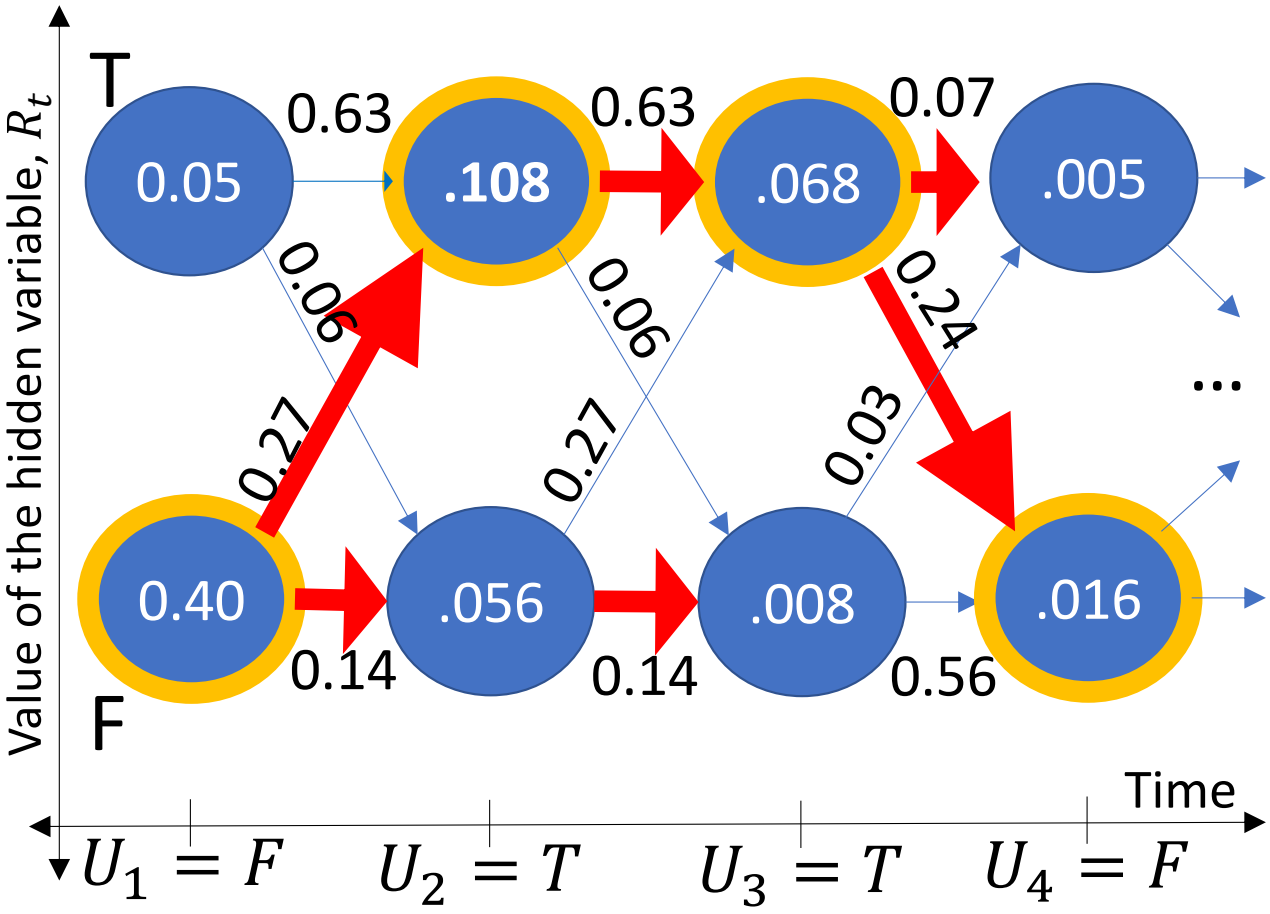
Follow the backpointers!

Given the observations

$$U_1, U_2, U_3, U_4 = F, T, T, F$$

… and given our hidden Markov model, we conclude that the most probable sequence of state variables is

$$R_1, R_2, R_3, R_4 = F, T, T, F$$



Some conclusions

- We have discovered that, if Elspeth brings her umbrella only on days 2 and 3, then the best inference is that it's raining on only those days.
- Well, this was kind of obvious from the beginning!

Some conclusions

Other types of HMMs might be less obvious. For example, consider the following assertion:

A fly flies well. A well does not fly.

In order to decide if these sentences are true or false, you first need to know which words are nouns, which verbs, and which adverbs.

In MP4, you will solve this problem using an HMM.

- State variable = part of speech
- Observation = word
- Transition model: verbs tend to come after nouns.

Final Word: Computational Complexity

- Inference by Enumeration in an HMM: $\mathcal{O}\{N^T\}$

$$P(\text{vars you care about}) = \sum_{\text{don't-care vars}} P(\text{all vars})$$

- Decoding using the Viterbi Algorithm: $\mathcal{O}\{TN^2\}$

$$v_{jt} = \max_i v_{i,t-1} e_{ijt}$$

Max over N values of i, performed for N values of j, and for T values of t.