CSCI 5622 Machine Learning

ML Sequence Learning

DATE	TOPIC	DUE
Mon, Dec 7	Sequence Learning	Peer Feedback Due
Wed, Dec 9	Reinforcement Learning	Peer Feedback Grades
Fri, Dec 11	----------	***Final Paper Due***
W 16th 4PM	***1777 Exposition Dr, Room at Entry***	Final Presentations
.. (Dec 16)	***Mandatory attendance & participation***	***Peer Project Questions***

www.RodneyNielsen.com/teaching/CSCI5622-F09/

Instructor: Rodney Nielsen

Assistant Professor Adjunct, CU Dept. of Computer Science

Research Assistant Professor, DU, Dept. of Electrical & Computer Engr.

Research Scientist, Boulder Language Technologies

Sequence Learning

- Sequences of commands
- Sequences of words in sentences
- Sequences of phonemes in spoken words
- Patterns of weather

Deterministic Sequences

- Traffic light
  - Always the same order
  - Easy to analyze once you know the transitions

Non-deterministic Patterns

ML

- Weather
  - Sunny, Cloudy, Stormy
  - but not necessarily in that order

Markov Assumption

ML

- Markov assumption
  - Assume the state depends only on the previous state
  - Simplifies problems significantly
  - May potentially be an oversimplification
  - E.g., today's weather depends only on the weather of the past few days, and not the wind, barometric pressure, etc.

Markov Process

ML

- State depends only on the previous *n* states
- Order *n* Markov model depends on *n* states
- A first order Markov model depends only on the previous state
  - With *n* states there are n^2 transitions
- State transitions are made probabilistically
  - State transition probabilities do not vary over time

Ex: State Transition Matrix

- Weather state transition matrix

		Today		
		sunny	cloudy	stormy
Yesterday	sunny	0.50	0.375	0.125
	cloudy	0.25	0.125	0.625
	stormy	0.25	0.375	0.375

- The sum of the probability of a row = 1.0
  - E.g., $P(Su_t|Su_{t-1}) + P(Cl_t|Su_{t-1}) + P(St_t|Su_{t-1}) = 1.0$
  - We know yesterday's weather and want to predict tomorrow's weather given that knowledge

Ex: State Transition Matrix

- Weather state transition matrix

		Today		
		sunny	cloudy	stormy
Yesterday	sunny	0.50	0.375	0.125
	cloudy	0.25	0.125	0.625
	stormy	0.25	0.375	0.375

- The sum of the probability of a column is not necessarily 1.0
  - E.g., $P(St_t|Su_{t-1}) + P(St_t|Cl_{t-1}) + P(St_t|St_{t-1}) = 1.125$
  - This is not what we are modeling

Initialization

- Must initialize the system according to the probable previous state
  - The π vector
  - E.g.,

sunny	cloudy	stormy
[0.8	0.1	0.1]

Markov process

- First order Markov process consists of:
  - States: (e.g., sunny, cloudy, stormy)
  - π vector: defines the state at time 0, initial state
  - State transition matrix: the probabilities of transitioning from one state to another
    - Fixed over the life of the model

Hidden States

- Assume you can't observe the weather, but can observe your dog
- Can we devise an algorithm to predict the hidden weather state based only on the state of the dog?

Speech Recognition

- Factors in interpreting speech
  - Vocal cords, throat, tongue, etc.
- Assume speech production is a sequence of hidden (internal) states
- The sound we hear is a sequence of observable states approximating the hidden states
  - E.g., there could be noise

Hidden vs. Observable States

ML

- The number of observable states might be very different than the number of hidden states
  - E.g., 3 hidden weather states (sunny, cloudy, stormy), but 4 states of the dog (exhausted, spunky, lazy, scared)
  - E.g., say 80 hidden (real) phoneme states but only 40 observable/distinguishable sounds

ML Hidden Markov Model (HMM)

- The hidden and observable states are probabilistically related

HMM Confusion Matrix

- Confusion Matrix: Gives the probability of an observable state given a hidden state

Hidden Weather State	Observable Dog State: exhausted	spunky	lazy	scared
sunny	0.60	0.20	0.15	0.05
cloudy	0.25	0.25	0.25	0.25
stormy	0.05	0.10	0.35	0.50

- Since we are assuming a given hidden state, the probability summed over a row is 1.0
  - E.g., $P(Ex_t|St_{t-1}) + P(Sp_t|St_{t-1}) + P(Lz_t|St_{t-1}) + P(Sc_t|St_{t-1}) = 1.0$

Hidden Markov Model (HMM)

- **Hidden states**: the *true* states of the system, the states we care about, are not observable
- **Observable states**: related states that are visible
- **π vector**: the initial probabilities of the hidden states
- **State transition matrix**: Gives the probability of moving from one hidden state to another
- **Confusion matrix**: Gives the probability of an observable state given a hidden state

Hidden Markov Model

- An HMM is a standard Markov process, where:
  - The true states are hidden,
  - But there are other observable states,
  - Which are probabilistically related to true states

Hidden Markov Model

- A Hidden Markov Model (HMM) is a triple (**π**, **A**, **B**)
  - **π** = (π_1): The vector of initial state probabilities
  - **A** = ($a_{i,j}$): The state transition matrix $P(x_{i,t} | x_{j,t-1})$
  - **B** = ($b_{i,j}$): The confusion matrix $P(y_i | x_j)$
- The probability matrices do not vary over time
  - Most unrealistic assumption about HMMs

HMM Uses

ML

- Evaluation: Compute the probability of a sequence of observed states given an HMM
- Decoding: Compute the probability of a sequence of hidden states given a sequence of observed states and the HMM

HMM Evaluation Task

- Suppose you want to determine which of a set of HMMs is responsible for generating a particular sequence of observations
  - E.g., you may have multiple weather models based on the season or multiple phoneme models based on the gender, accent, or other attributes of the speaker

HMM Evaluation Task

ML

- Suppose you want to determine which of a set of HMMs is responsible for generating a particular sequence of observations
  - For each HMM, compute the probability of the given sequence of observed states using the forward algorithm
  - Select the HMM that is the most probable

HMM Evaluation Task

ML

- Suppose you want to determine which of a set of HMMs is responsible for generating a particular sequence of observations
- E.g., speech recognition
  - Start with several word models (HMMs)
  - Record a sequence of observed sounds
  - Determine word by computing most probable HMM to have generated it

HMM Decoding Task

ML

- What sequence of hidden states generated the sequence of observations
- Compute the most probable sequence of hidden states given an HMM and a sequence of observations using the Viterbi algorithm

Ex: HMM Decoding Task

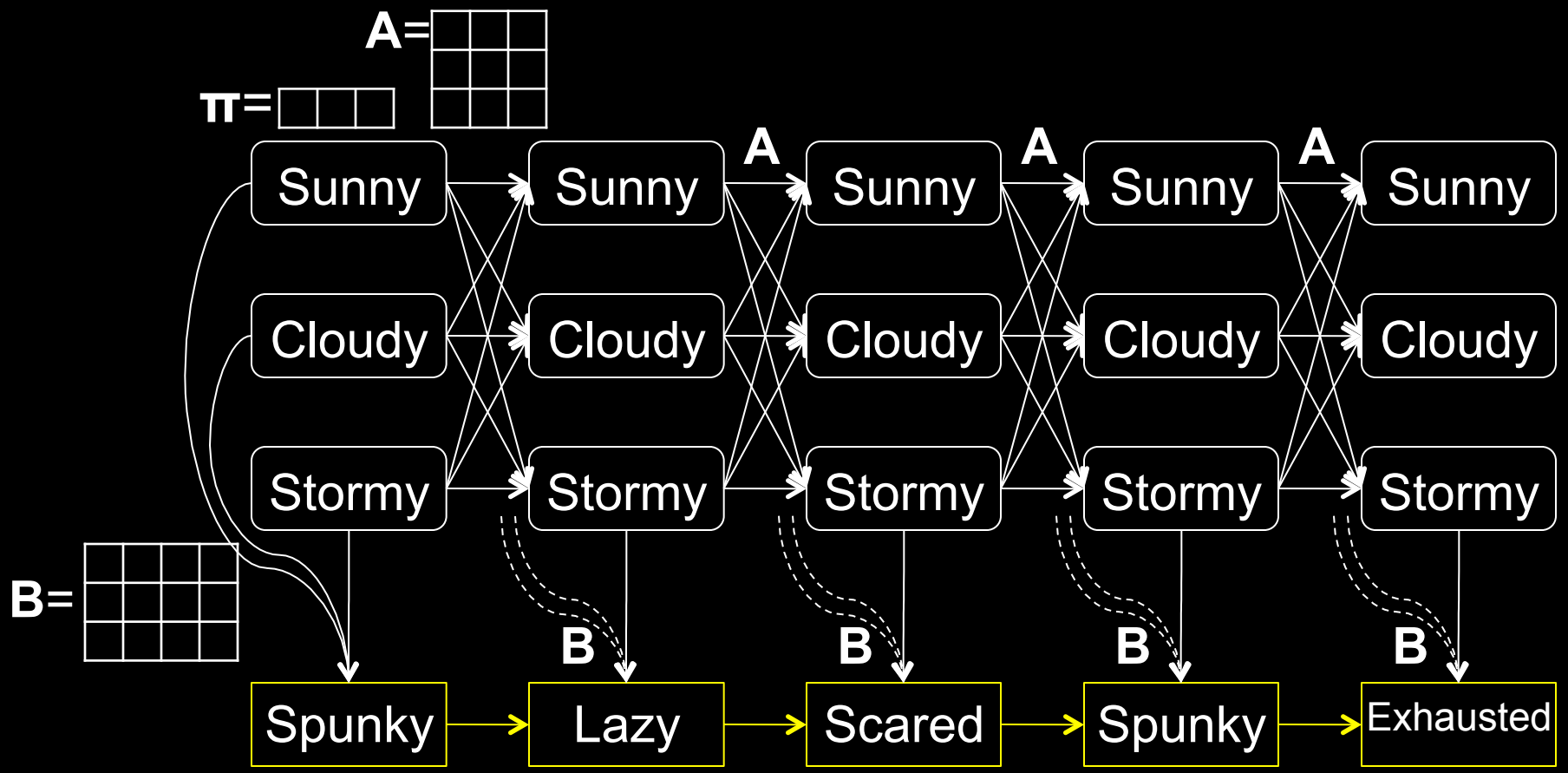
ML

- What sequence of hidden part-of-speech (POS) states generated the sequence of observed word states
  - She was busily decoding$_{\text{verb}}$ the message.
  - The decoding$_{\text{common-noun}}$ was "*good luck!*".
  - Decoding$_{\text{proper-noun}}$ Systems, Inc. has all the fun.
  - The decoding$_{???}$ task was fun!

Learning an HMM

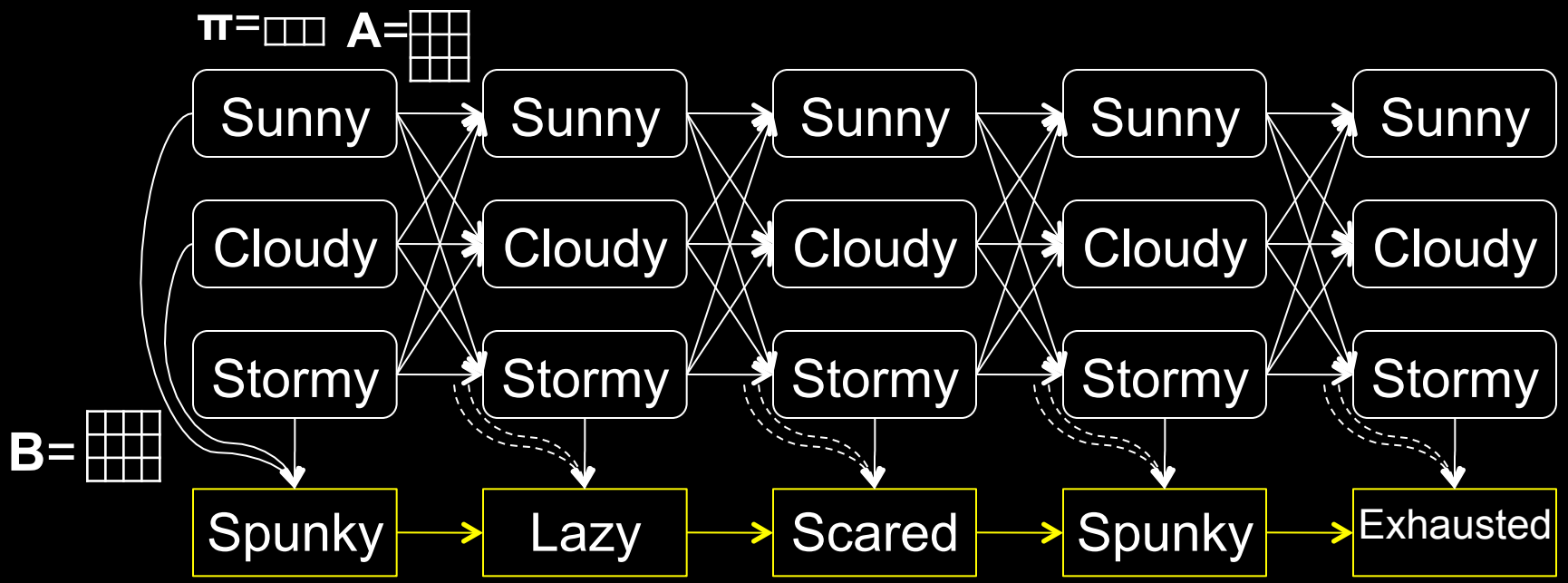
- Learn the most probable HMM, ($\boldsymbol{\pi}$, **A**, **B**), given a set of observed state sequences associated sequences of underlying hidden states
- Use the forward-backward algorithm to learn the matrices **A** and **B**
  - These probabilities are often unknown in real-world scenarios

ML Evaluation: *P*(Observed Sequence)



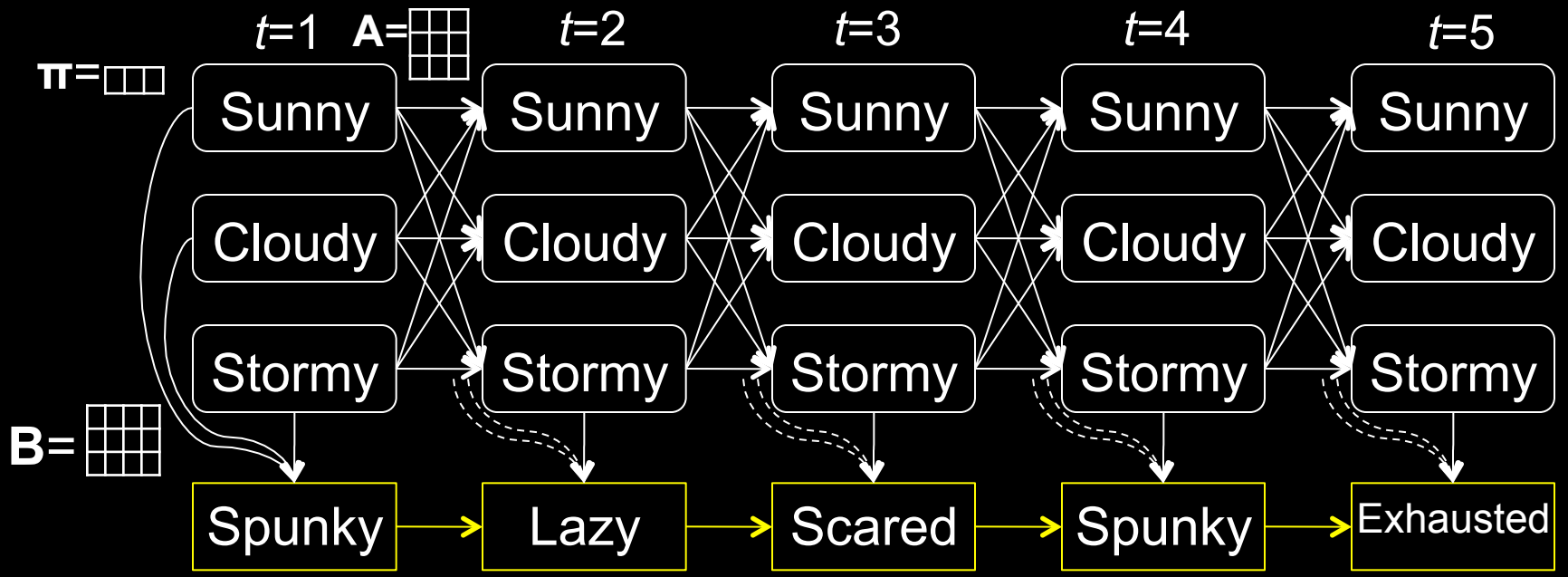
Exhaustive Search

- P(Sp,Lz,Sc,Sp,Ex) = P(Sp,Lz,Sc,Sp,Ex|Su,Su,Su,Su,Su) +
  P(Sp,Lz,Sc,Ex|Su,Su,Su,Su,Cl) + … + P(Sp,Lz,Sc,Sp,Ex|St,St,St,St,St)
- **Too Expensive**



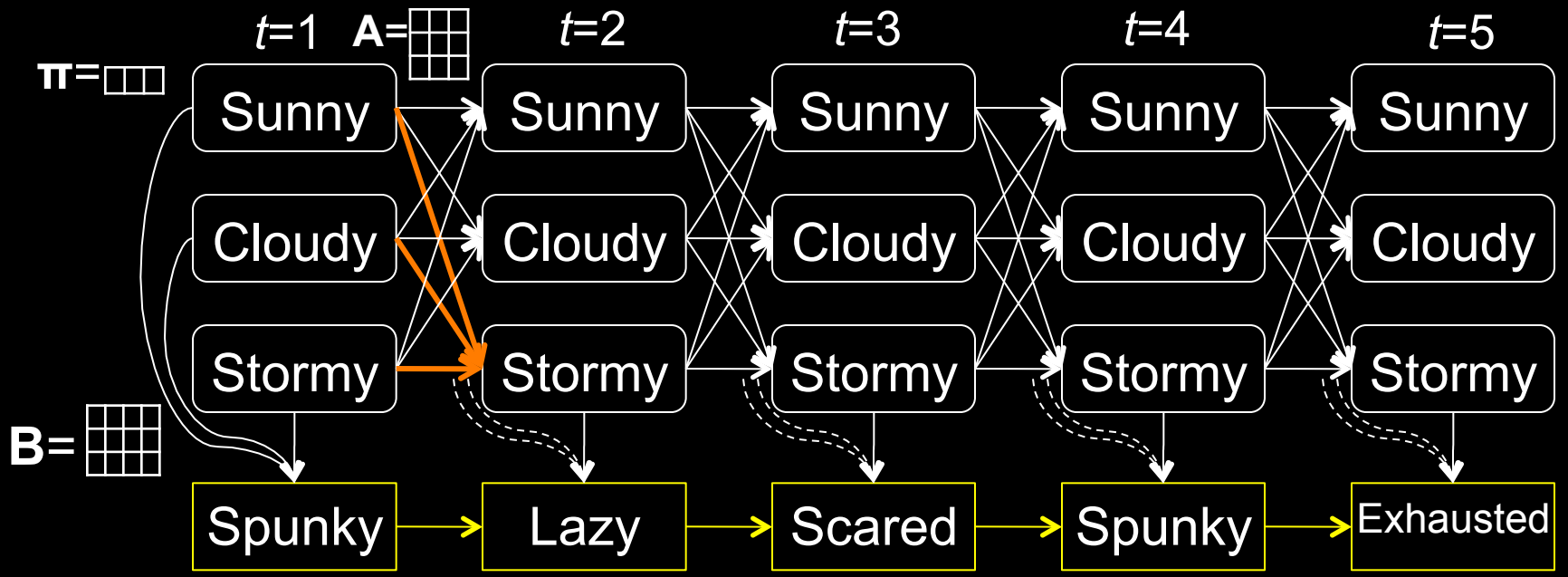
Forward Algorithm

- Instead calculate the probabilities recursively
- Consider the probability of reaching an intermediate state in the trellis, say the probability that it is Stormy at *t*=2



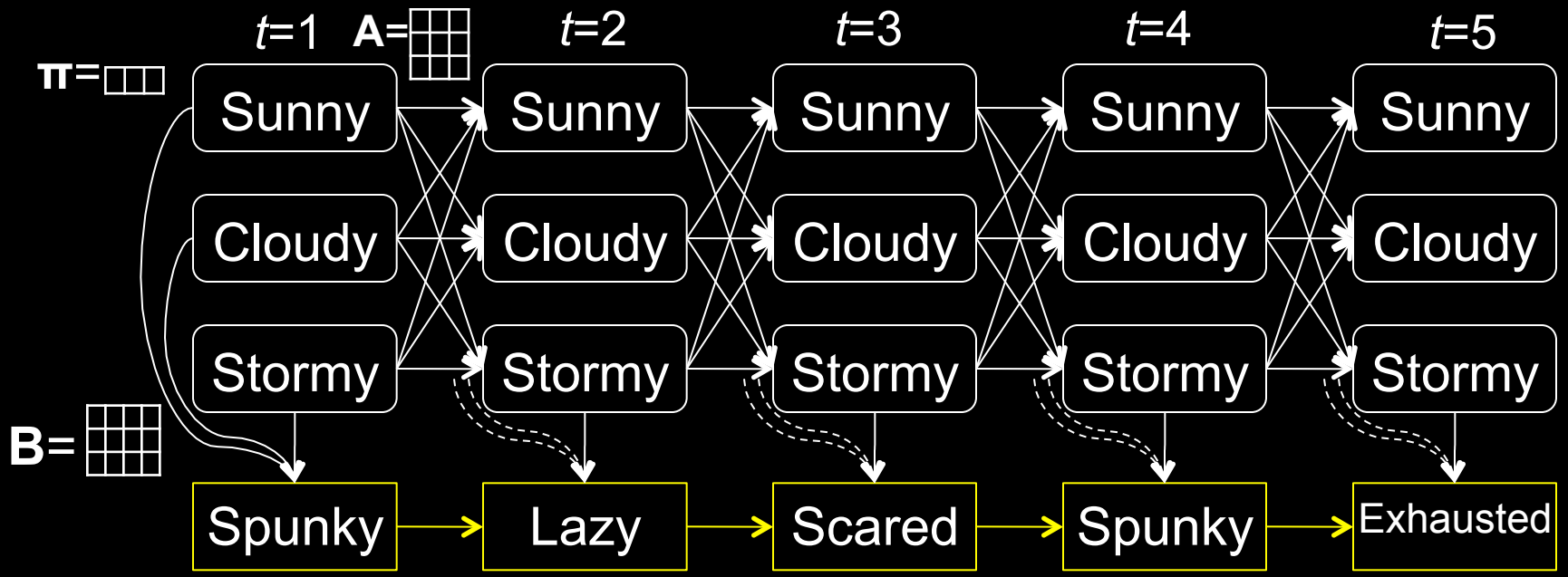
Forward Algorithm

- Let $\alpha(j_t)$ be the prob of reaching state j at tm t
- $\alpha(j_t) = P(obs=y_t|j_t) \times \Sigma_{path\ to\ j_t} P(path)$
- $\alpha(St_2) = P(Lz_2|St_2) \times P(paths\ below\ in\ orange)$



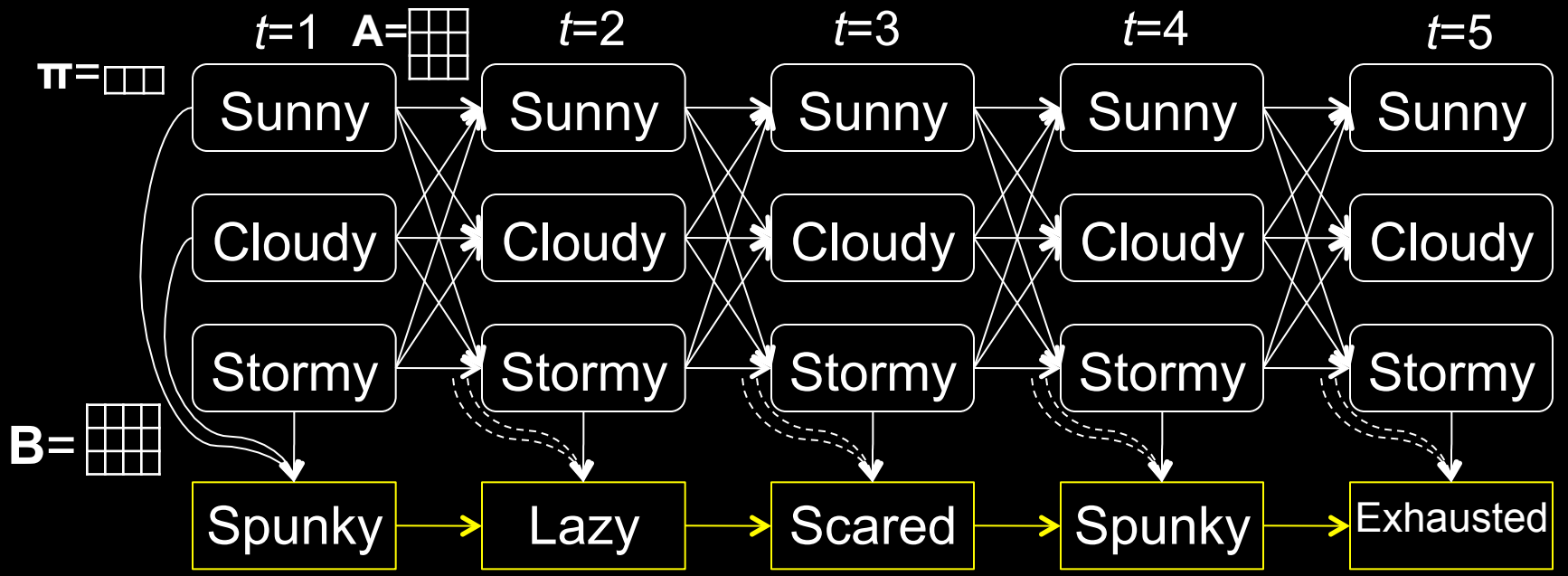
Forward Algorithm

- P(Sp,Lz,Sc,Sp,Ex)
  = The sum of the probabilities over all paths
  = The sum of the final partial probabilities $\alpha(j_5)$



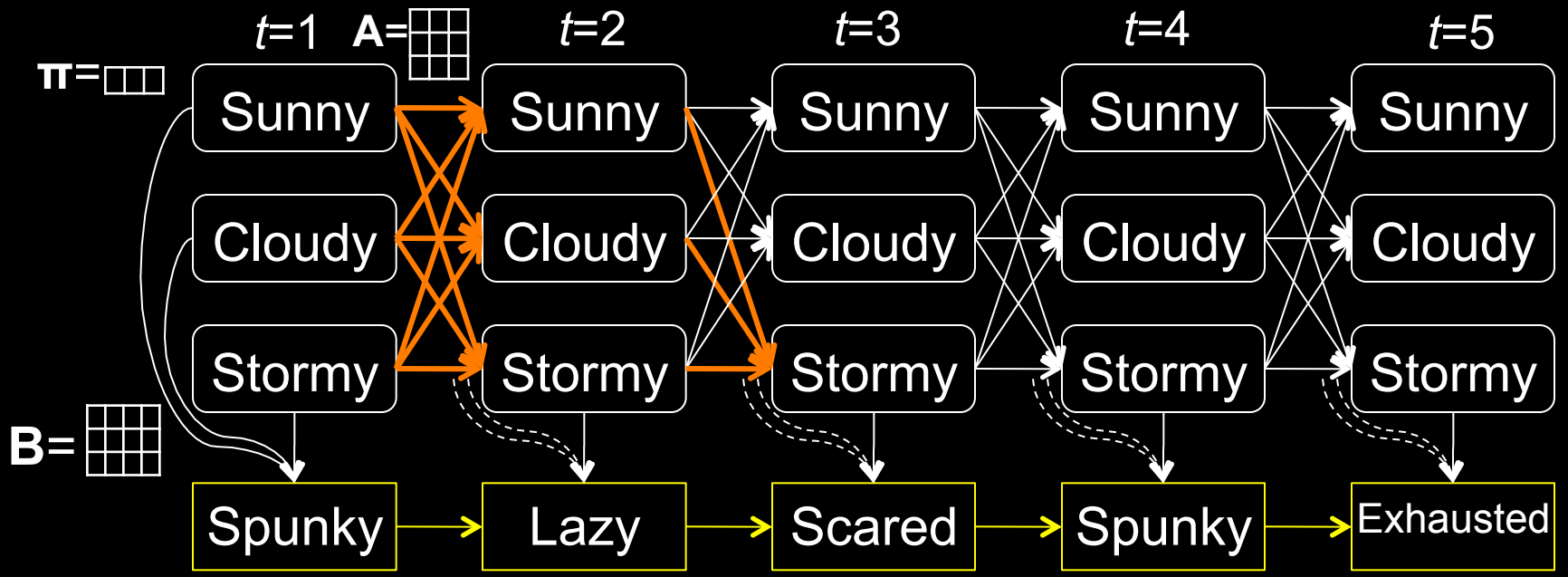
Forward Algorithm

- $\alpha(j_t) = P(obs{=}y_t | j_t) \times \Sigma_{path\ to\ j_t} P(path)$; $\alpha(j_1) = ?$
- $\alpha(j_1) = P(y_1 | j_1) \times \Sigma_{path\ to\ j_t} P(path)$
  $= b_{j,y_1} \times \pi(j)$



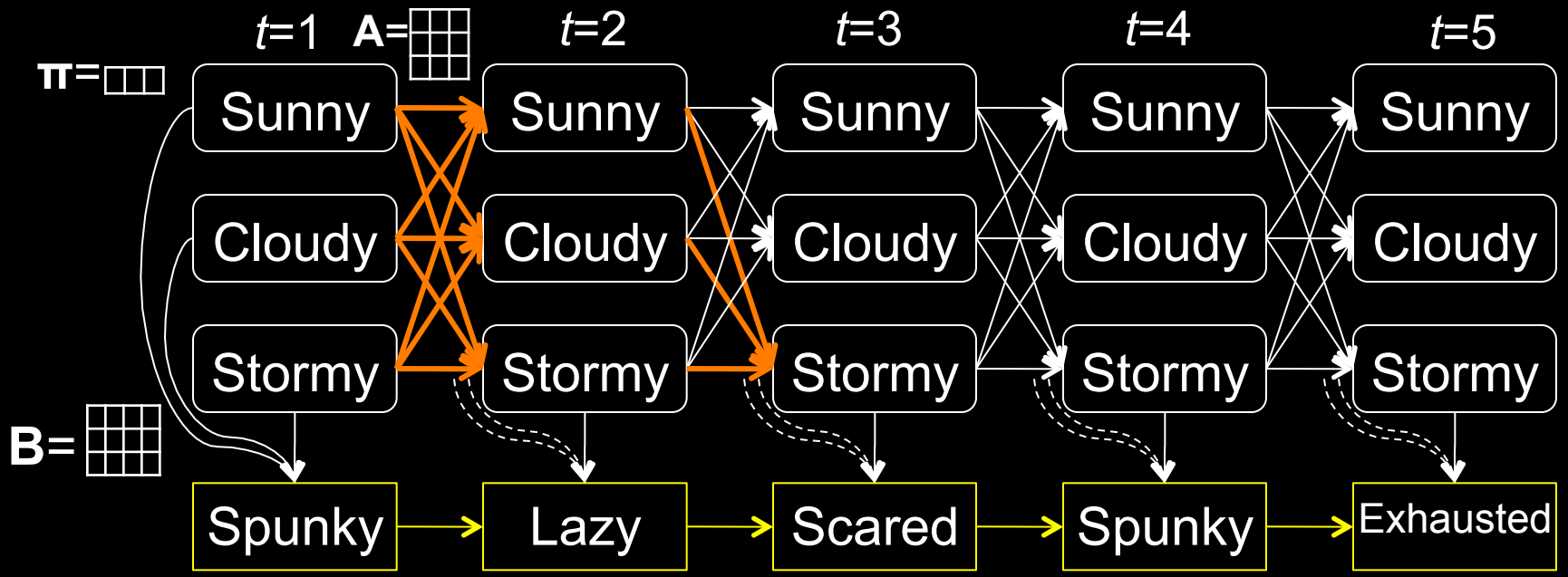
Forward Algorithm

- $\alpha(j_t) = P(obs{=}y_t | j_t) \times \Sigma_{path\ to\ j_t} P(path)$
- $\alpha(St_3) = P(Sc_3 | St_3) \times \Sigma_{9\ paths\ to\ St_3} P(path)$
  $= b_{Sc,St} \times \Sigma_{9\ paths\ to\ St_3} P(path)$



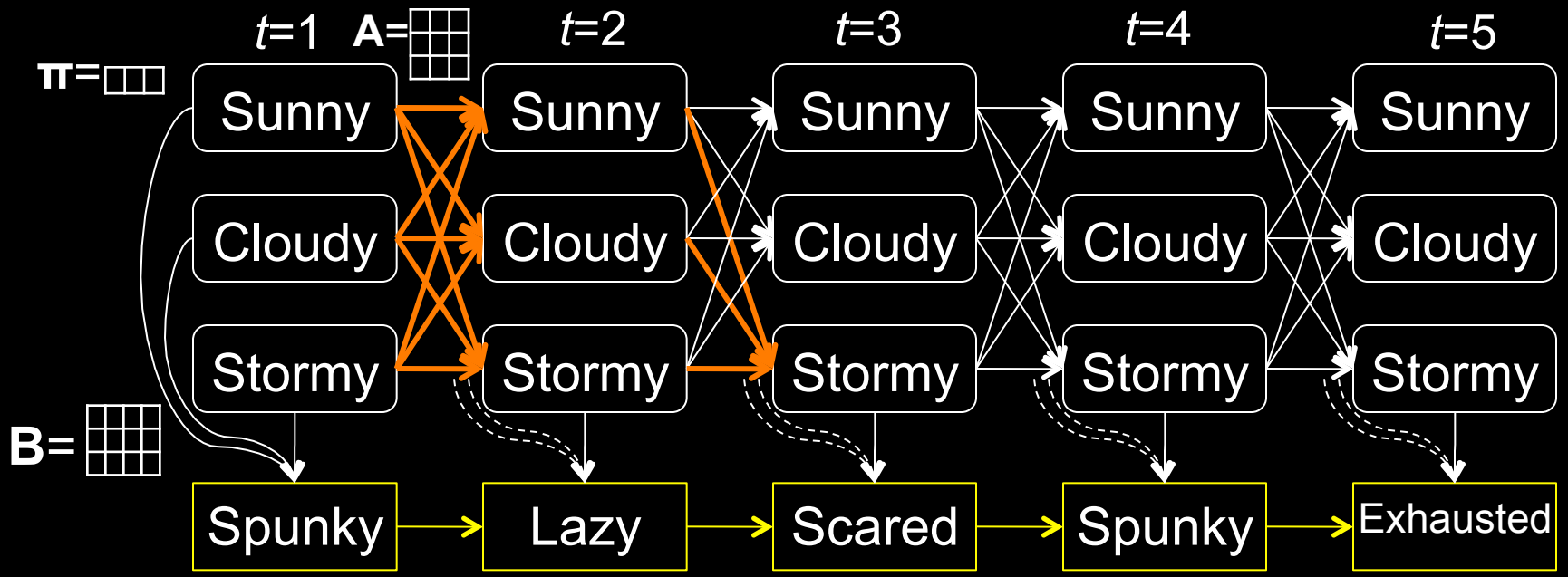
Forward Algorithm

- The number of paths grows exponentially with the length of the observation sequence, but…
- We can recursively compute the $\alpha(j_{t-1})$



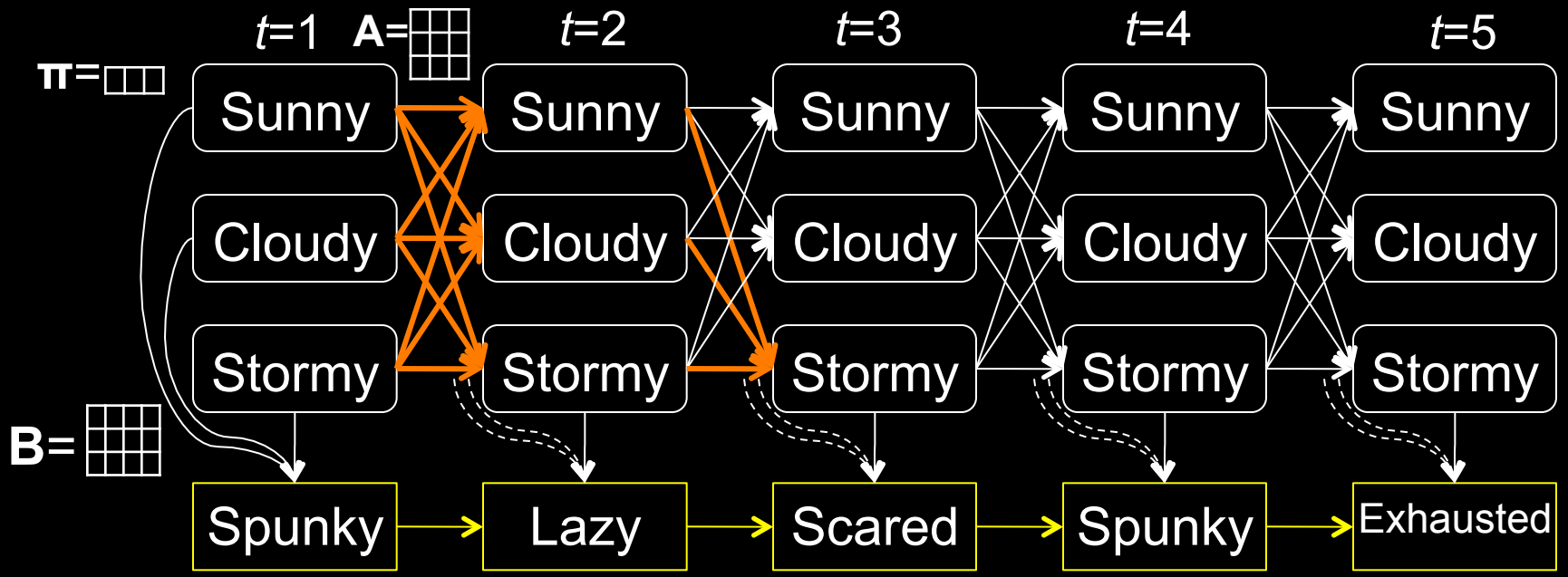
Forward Algorithm

- $\alpha(j_t) = b_{y,j} \times \Sigma_i \alpha(i_{t-1}) a_{i,j}$
- Compute the $\alpha(j_t)$ forward from t=1 to t=T
- The $\Sigma_j \alpha(j_T)$ is the probability of the sequence



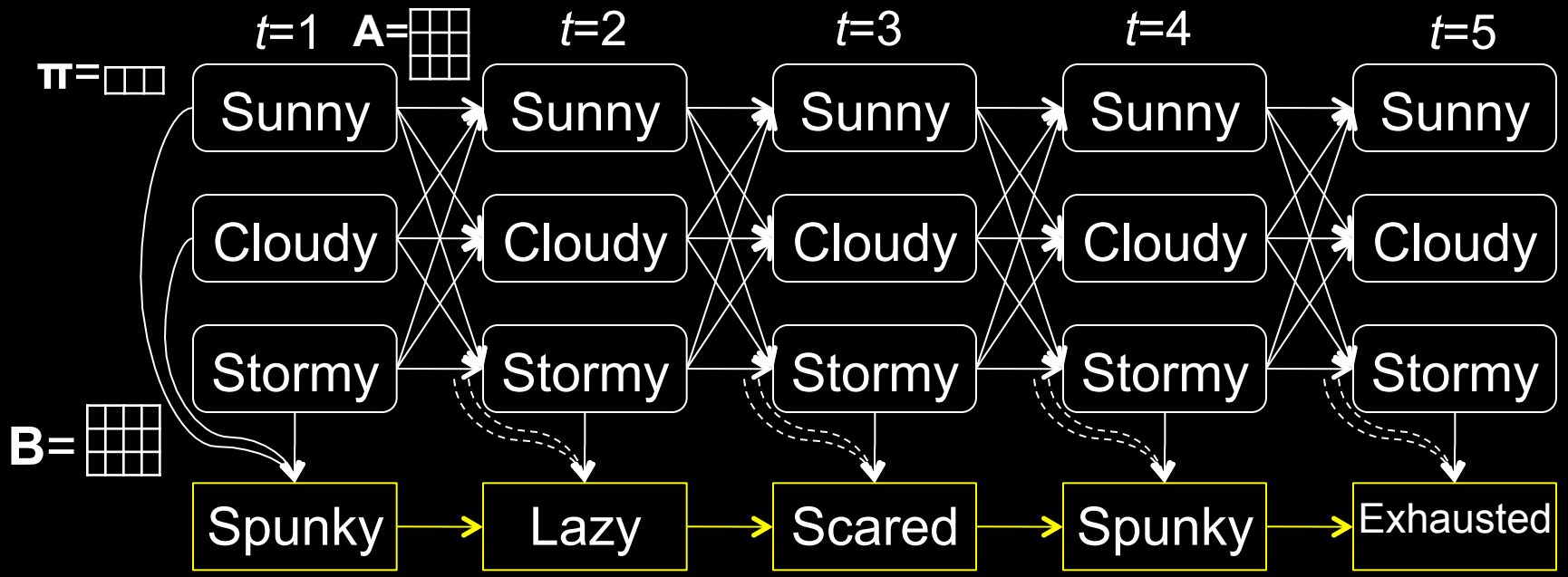
Forward Algorithm

- Exhaustive search is $O(n^T)$
- Forward search is $O(nT) << O(n^T)$



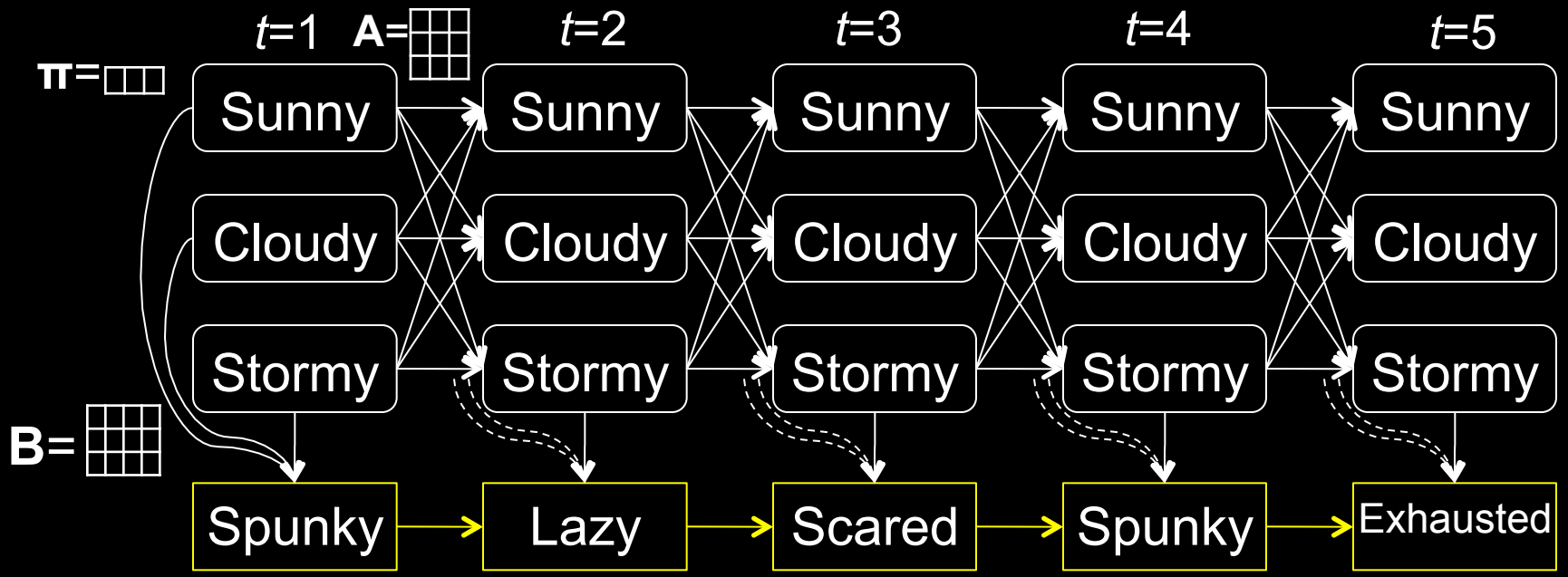
The Sequence of Hidden States

- Finding the most probable sequence of hidden states
- Exhaustive search: enumerate all the possible state paths and their probabilities
- Select the path that maximizes the probability



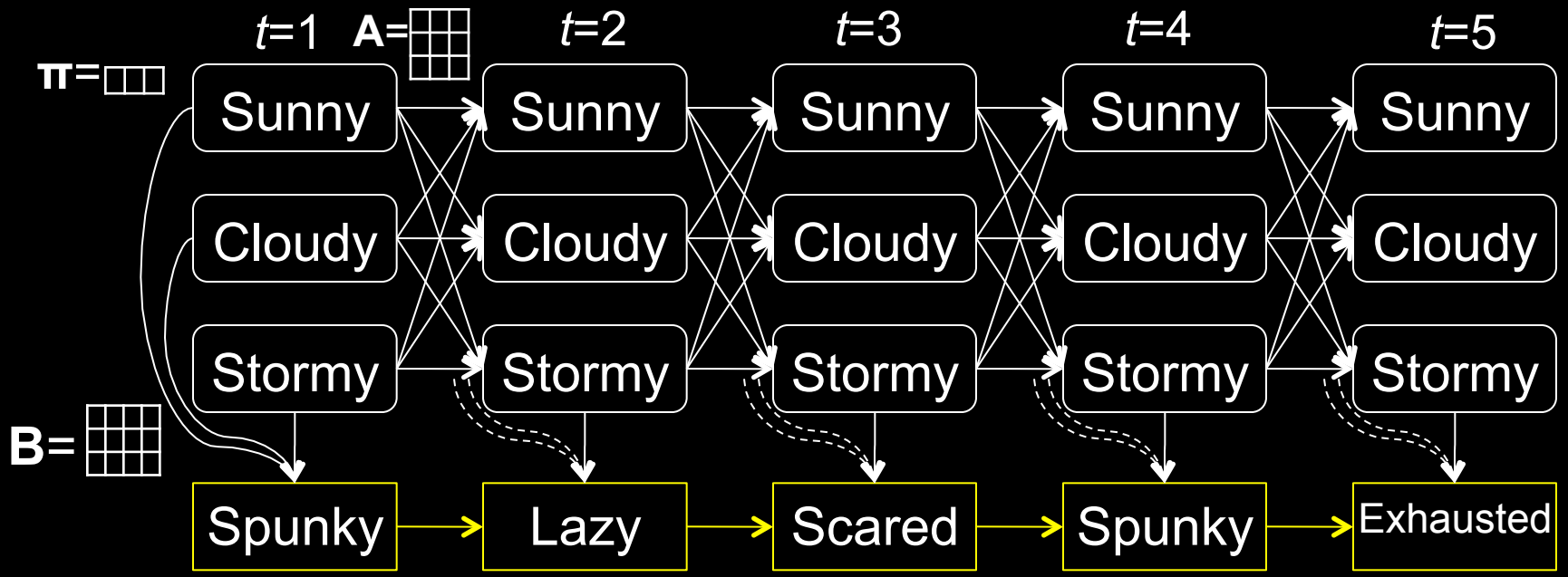
The Sequence of Hidden States

- Finding the most probable sequence of hidden states
- Exhaustive search: enumerate all the possible state paths and their probabilities: $O(n^T)$, **too expensive**
- Select the path that maximizes the probability



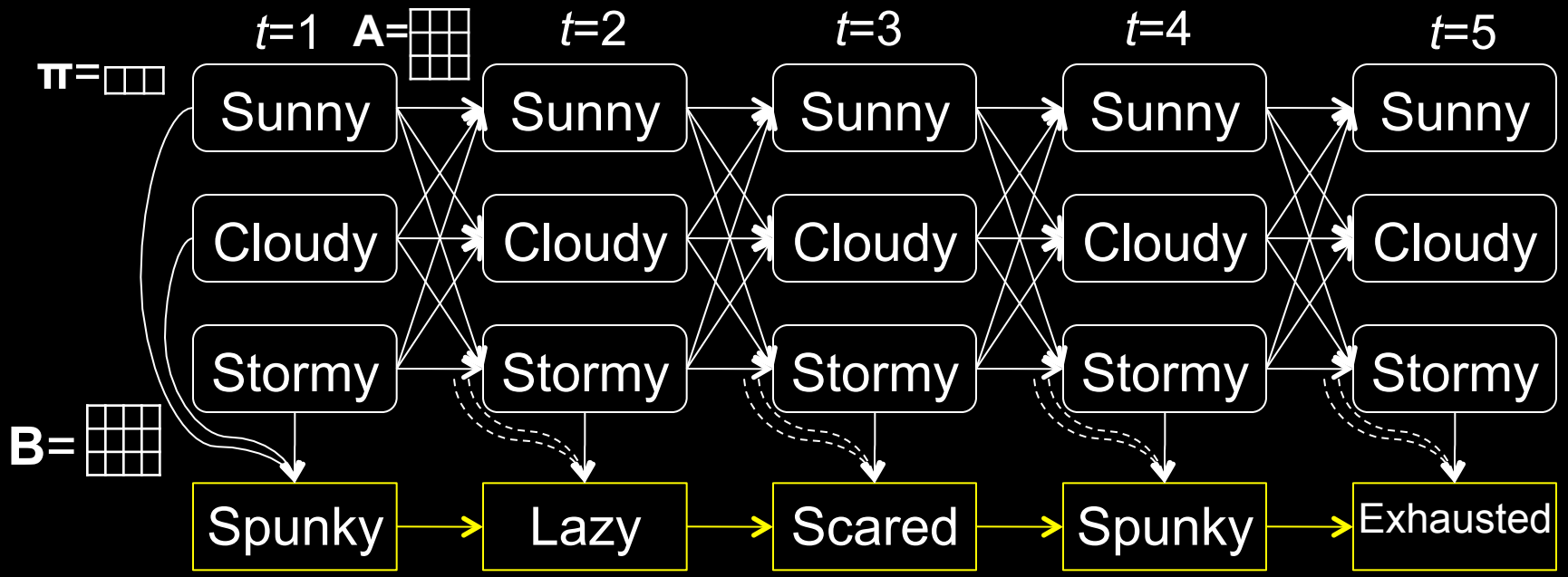
Viterbi Algorithm

- Now, we want the highest probability path versus the sum of the probability over all paths
- Each state has a most probable path to it



Viterbi Algorithm

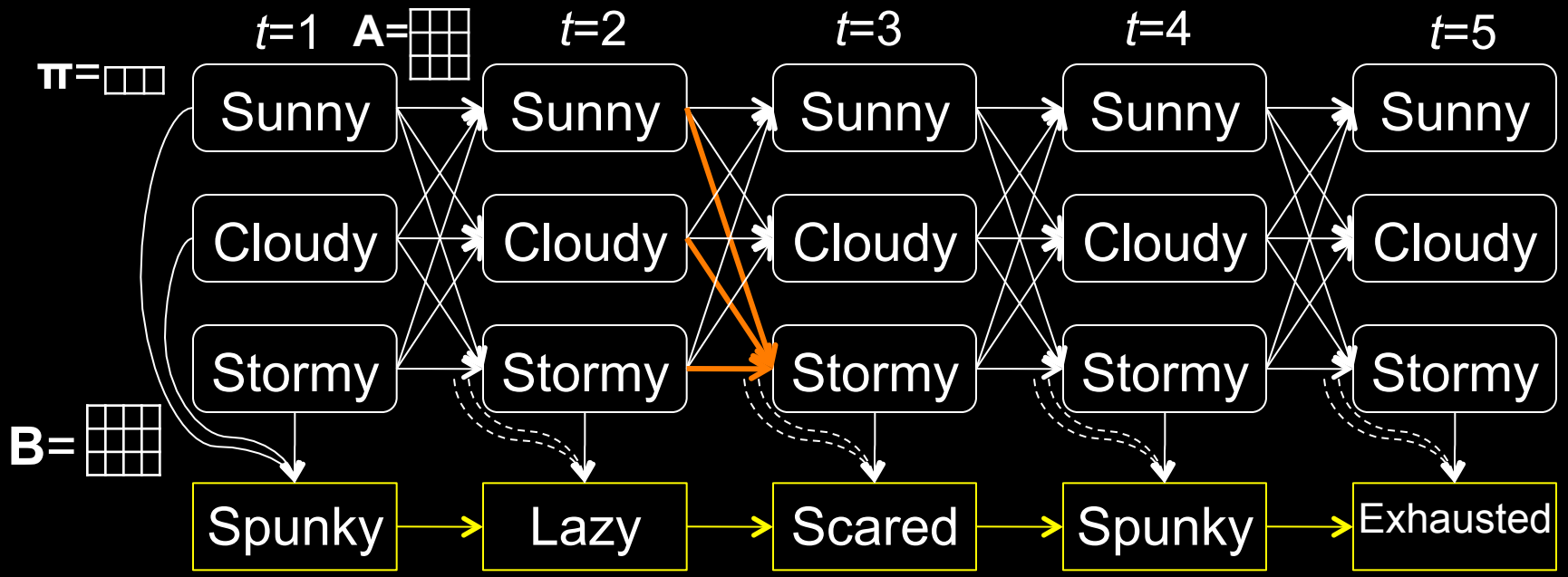
- Find most probable path to state j_T
- Choose the state with the maximum partial probability and use its partial best path



Viterbi Algorithm

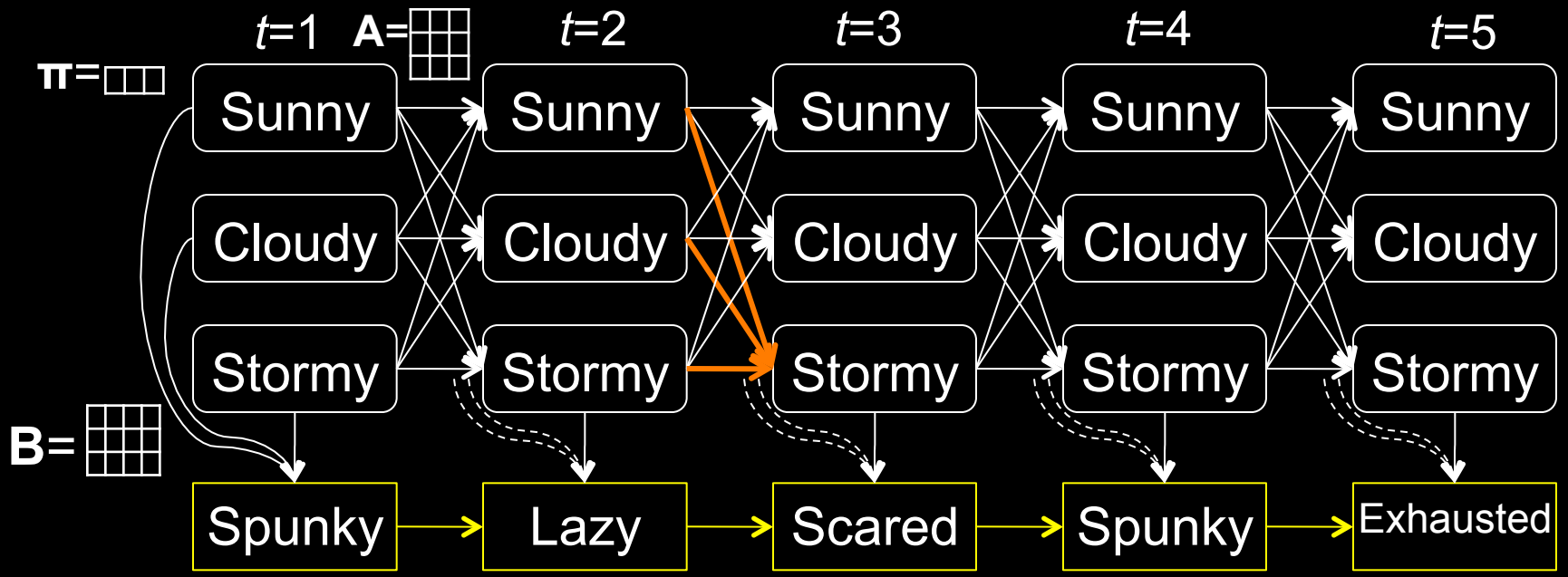
ML

- $\delta(i_1) = \pi(i) b_{i,k}$
- The most probable path to St_3 is the max of:
- P(Most probable path to j_2)$P(St|j)$



Viterbi Algorithm

- So the probability of the most probable path is:
- $\delta(i_t) = \max_j \delta(j_{t-1}) a_{j,i} b_{i,y_t}$



Questions

ML

- Questions???

Conference Deadlines

ML

- AAAI
  - http://www.aaai.org/Conferences/AAAI/aaai10.php
  - January 18, 2010: Electronic abstracts due
  - January 21, 2010: Electronic papers due

ML Presentations

- Mon, Dec 7: Nirav & Kun
- Wed, Dec 9: Tony & Xinyu

Projects

ML

- Will stay until all questions are addressed, or
- Please make an appointment if you have any questions at all regarding your project