NPFL099 - Statistical dialogue systems

Dialogue management

Belief monitoring I

Filip Jurčíček

Institute of Formal and Applied Linguistics
Charles University in Prague
Czech Republic

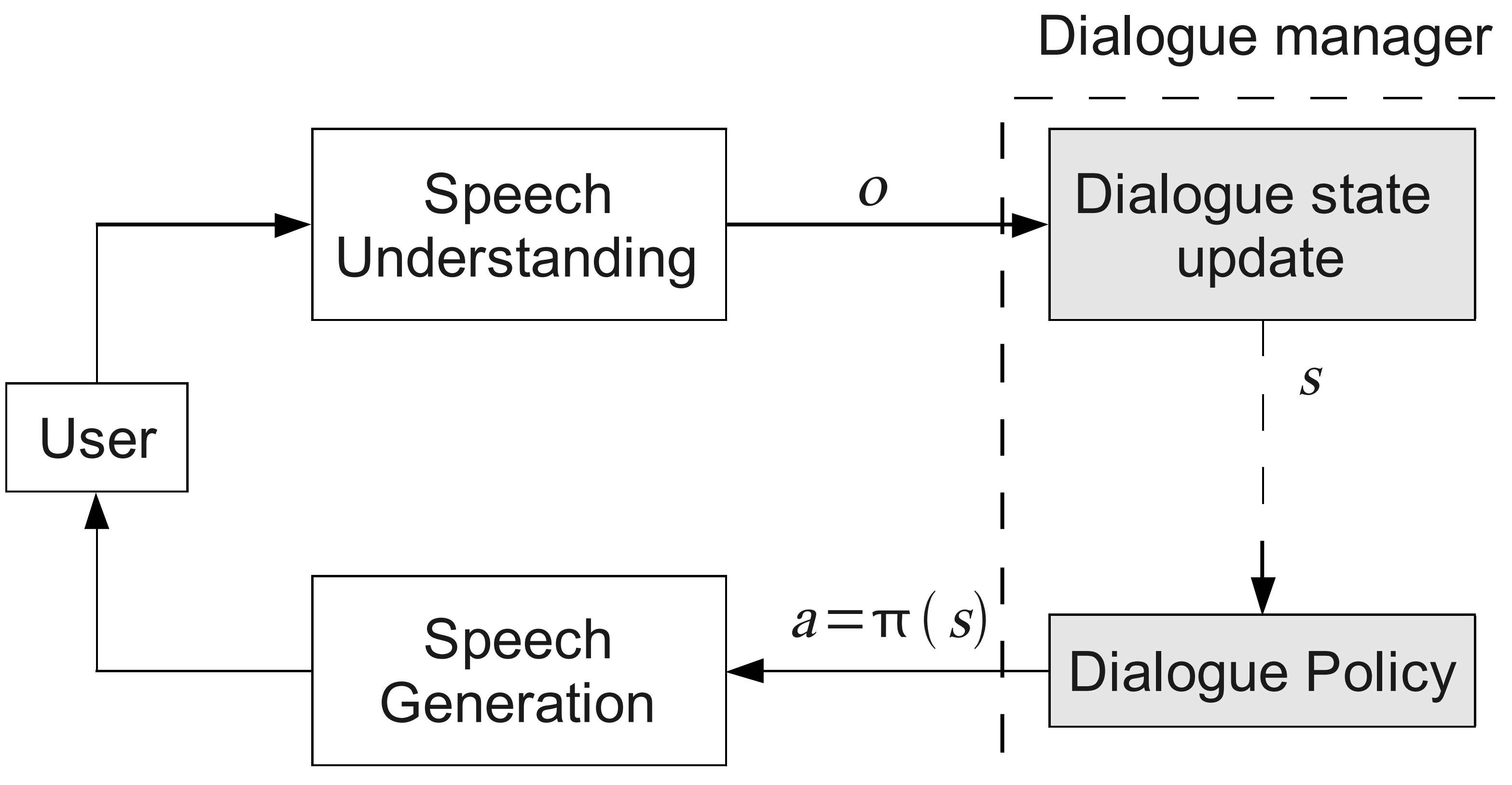
Home page: http://ufal.mff.cuni.cz/~jurcicek

Version: 20/03/2013

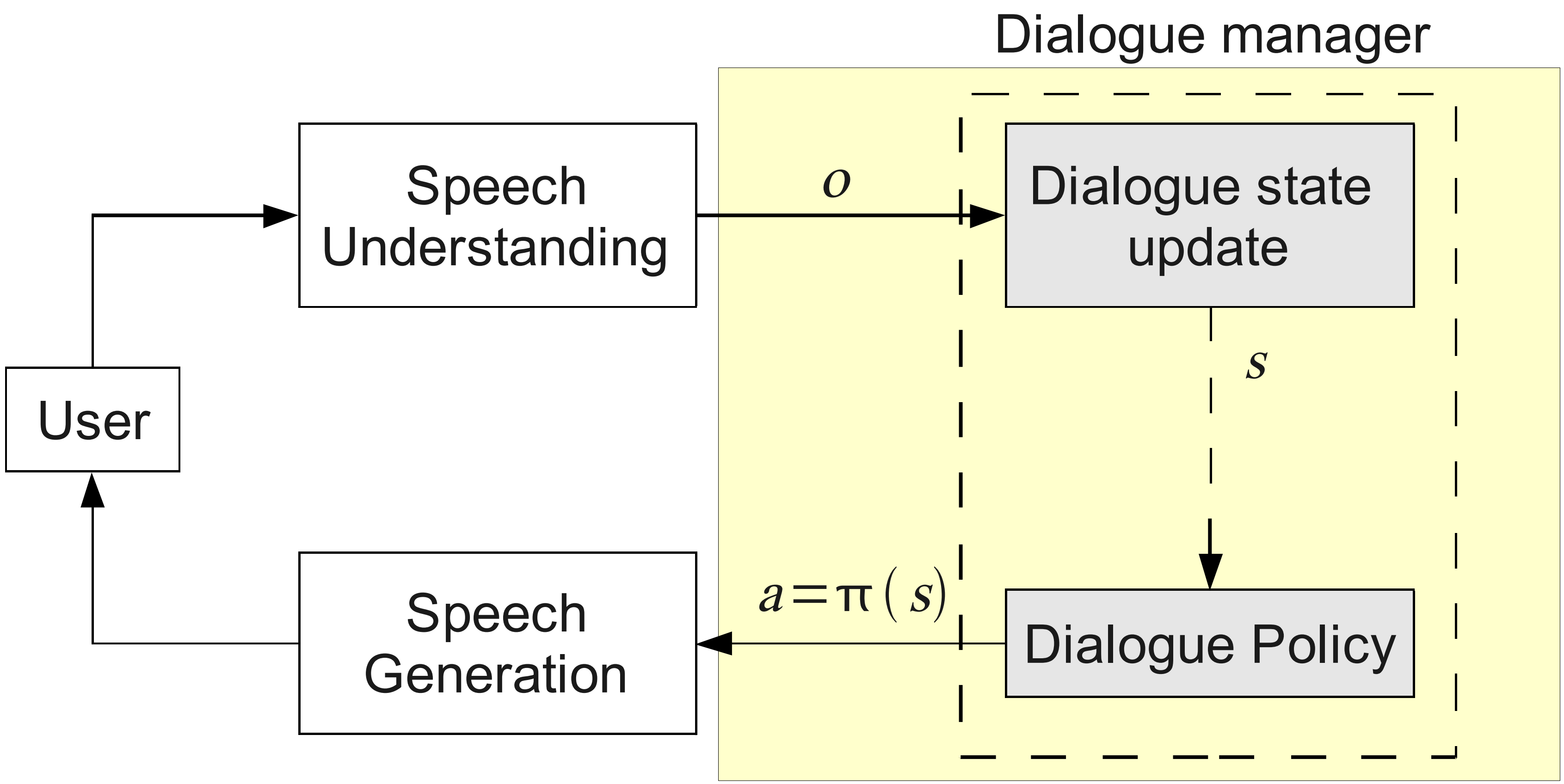
Outline

- What is a dialogue manager?
- Dialogue state definition
- Motivation for statistical SDS
- Dialogue state estimation
- State enumeration and pruning

Typical spoken dialogue systems



Dialogue state and policy



- Dialogue state is composed of **variables** needed to track the the progress of the dialogue
- Policy is implemented as a sequence of **if/then decision**

Example: TownInfo application

- Queries about
  - restaurants, bars, and hotels
- Search constraints
  - area, price range, stars
- Provides
  - address, postcode, phone number

- BUDS by B. Thomson, CAM, UK
  - Call (22191) 9888

- ALEX by DSG, UFAL, CZ ;-)
  - Call (22191) 9889

Example of conversation

Turn	Transcription	Dialogue act
System	Hello. How may I help you?	hello()
User	Hi, I am looking for a restaurant.	inform(venue_type=restaurant)
System	What type of food would you like?	request(food_type)
User	I want Italian.	inform(food_type=Italian)
System	Did you say Italian	confirm(food_type=Italian)

Real user input

User	0.4 hi I am looking for a restaurant 0.2 uhm am looking for a bar	0.7 inform(venue_type=restaurant) 0.3 inform(venue_type=bar)
System	Did you say that you are looking for a restaurant?	confirm(venue_type=restaurant)

Dialogue state

Dialogue state is used to track the progress of the dialogue

E.g. a set of random variables:

- venue_type
- food_type
- price_range
- area
- stars

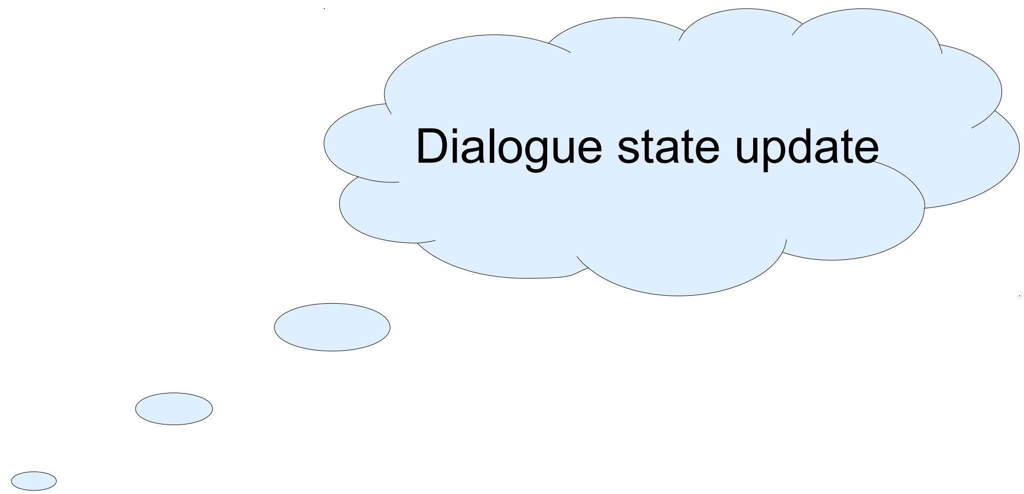
User says

Dialogue state is used to track the progress of the dialogue

- Turn 1:
  - S: How may I help you?
  - Dialogue state:
    - venue_type = None
    - food_type = None
    - price_range = None
  - U: inform(venue_type = restaurant)

Dialogue state update

Dialogue state is used to track the progress of the dialogue



- Turn 1:
  - S: How may I help you?
  - Dialogue state:
    - venue_type = restaurant
    - food_type = None
    - price_range = None
  - U: inform(venue_type = restaurant)

System says

Dialogue state is used to track the progress of the dialogue

- Turn 2:
  - S: What type of food are you looking for?

  - Dialogue state:
    - venue_type = restaurant
    - food_type = None
    - price_range = None

User says

Dialogue state is used to track the progress of the dialogue

- Turn 2:
  - S: What type of food are you looking for?
  - Dialogue state:
    - venue_type = restaurant
    - food_type = None
    - price_range = None
  - U: inform(food_type=Chinese)

Dialogue state update

Dialogue state is used to track the progress of the dialogue

- Turn 2:
  - S: What type of food are you looking for?
  - Dialogue state:
    - venue_type = restaurant
    - food_type = Chinese
    - price_range = None

Dialogue state update

Ontology

- Used to define the structure of a dialogue state and dependencies between variables
- It can simplify building a new SDS for a new domain

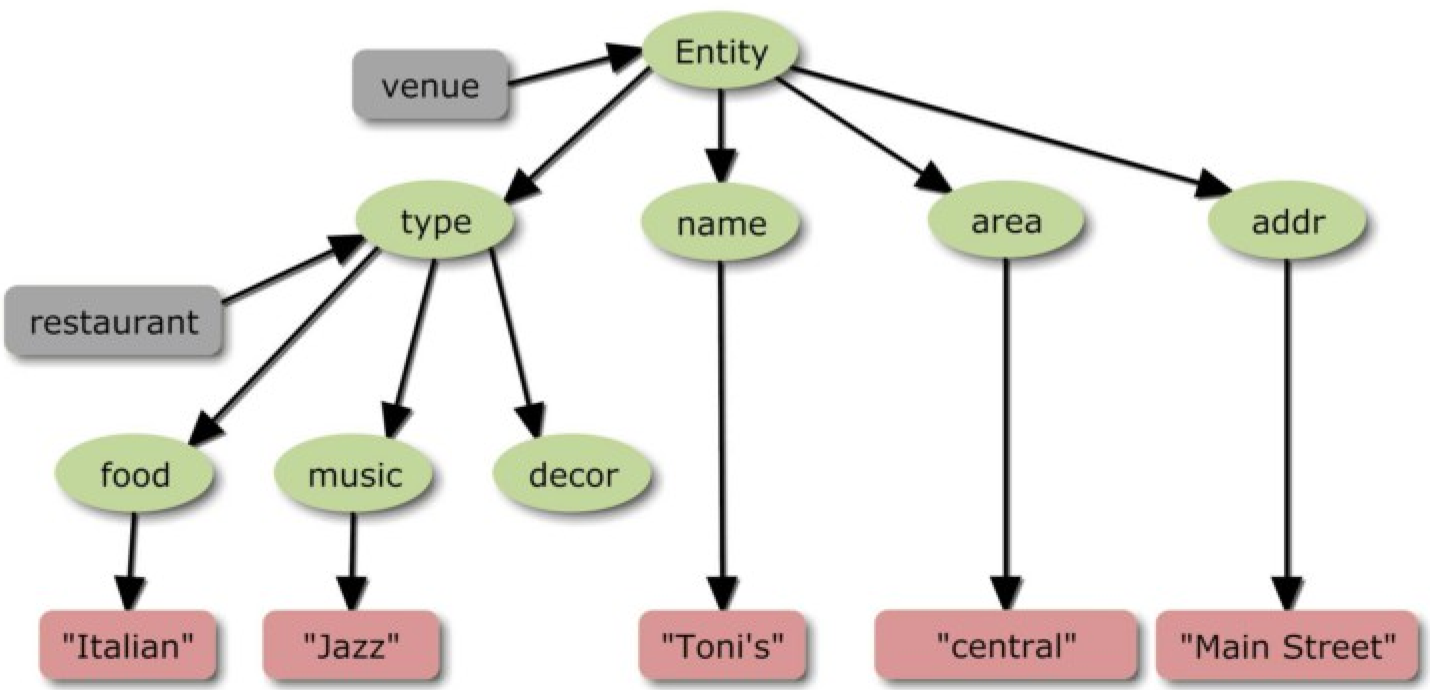


Figure 2: Example Tree for TownInfo Application

Ontology

```
# top level tasks
task -> find (-entity, -method,
-discourseAct);

# define main entities in domain
entity -> venue(type, +area, +near,
-addr, -phone, -postcode, *reviews,
*rating, +pricerange, -price);

# places to eat
type -> restaurant(+food);
type -> pub(childrenallowed,
 hasinternet, hastv);
type -> coffeeshop(food);

# attributes
pricerange = ( "free" | "cheap"
  | "moderate" | "expensive");

area = (girton|kingshedges|arbury|
  ...
  citycentre|riverside|castlehill);
```

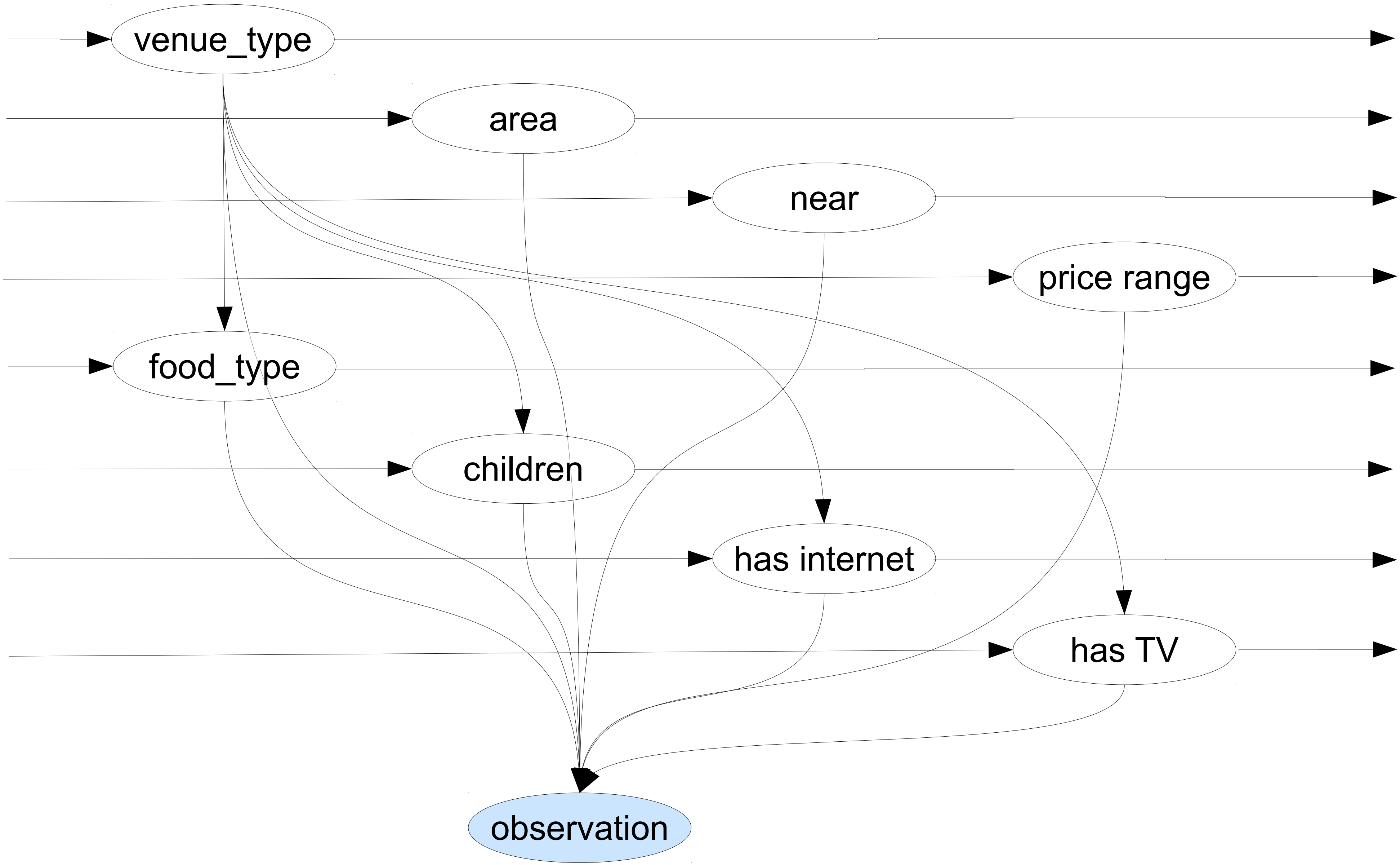
```
food = ( "American"
  ...
  | "Chinese takeaway");

hasinternet = ( true | false);
hastv = ( true | false);
childrenallowed = ( true | false);

# descriptive lexical types

addr = ();
phone = ();
postcode = ();
price = ();
rating = ();
reviews = ();
```

Ontology

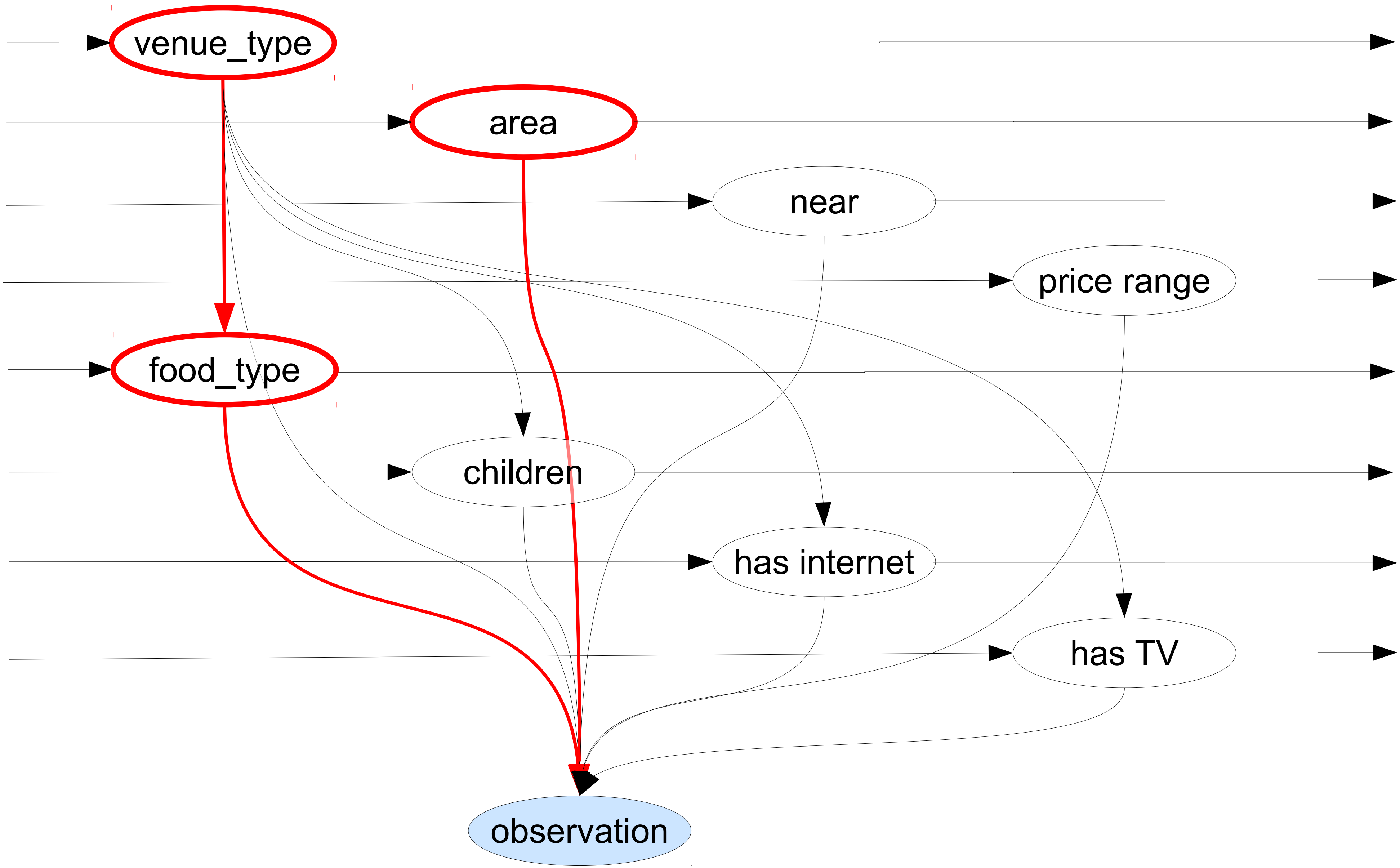
- Defines
  - all concepts in the domain
  - their dependencies
  - should be requested by the system (+)
  - should not be requested by the system (-)

- Dependencies:
  - some concepts are applicable only for some values of parents concepts' values
  - e.g.
    - hastv only if venue_type = pub
    - food only if venue_type = restaurant

TownInfo influence network

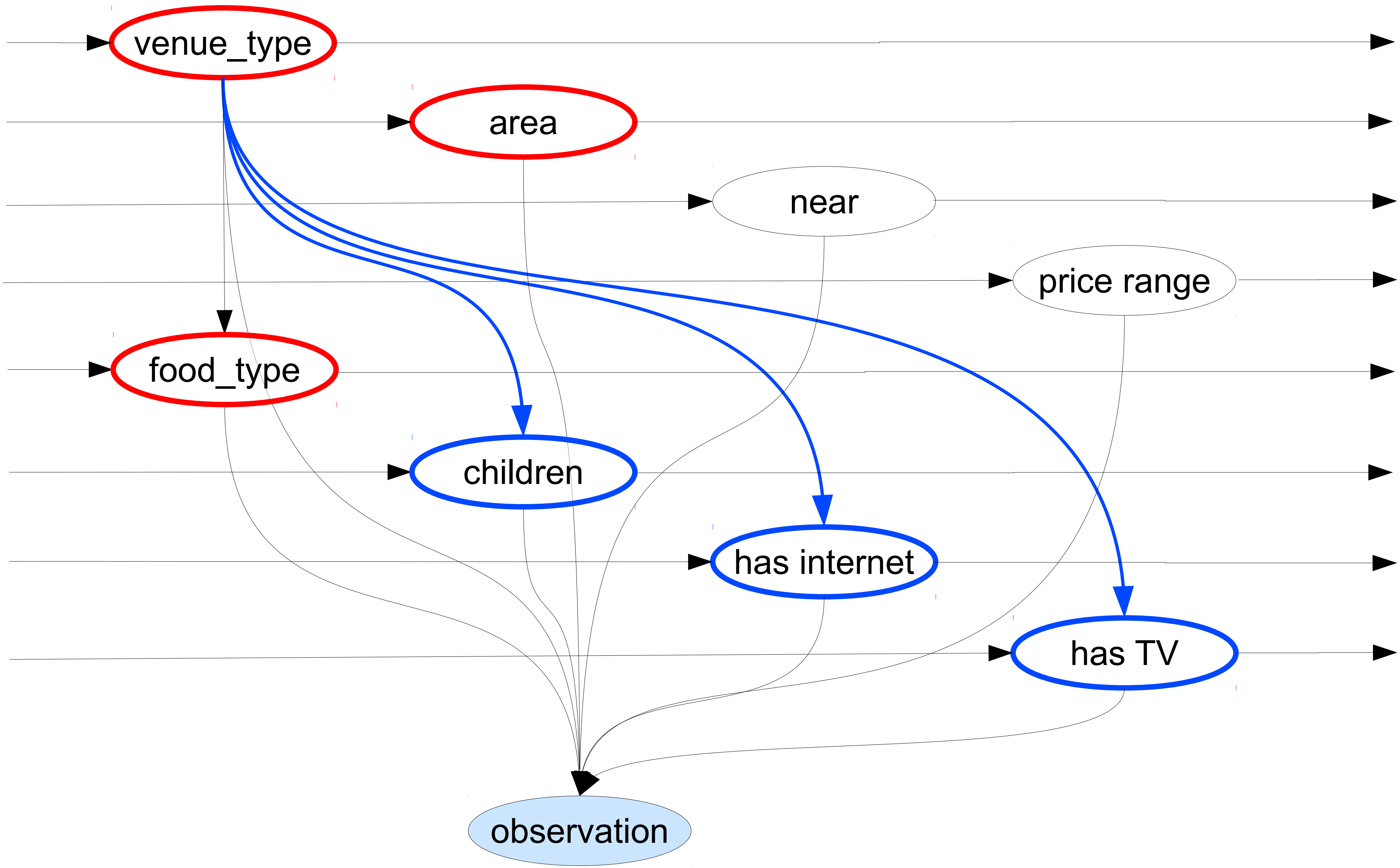


Store new information



inform(food_type=Chinese)&inform(area=centre)

Update old information

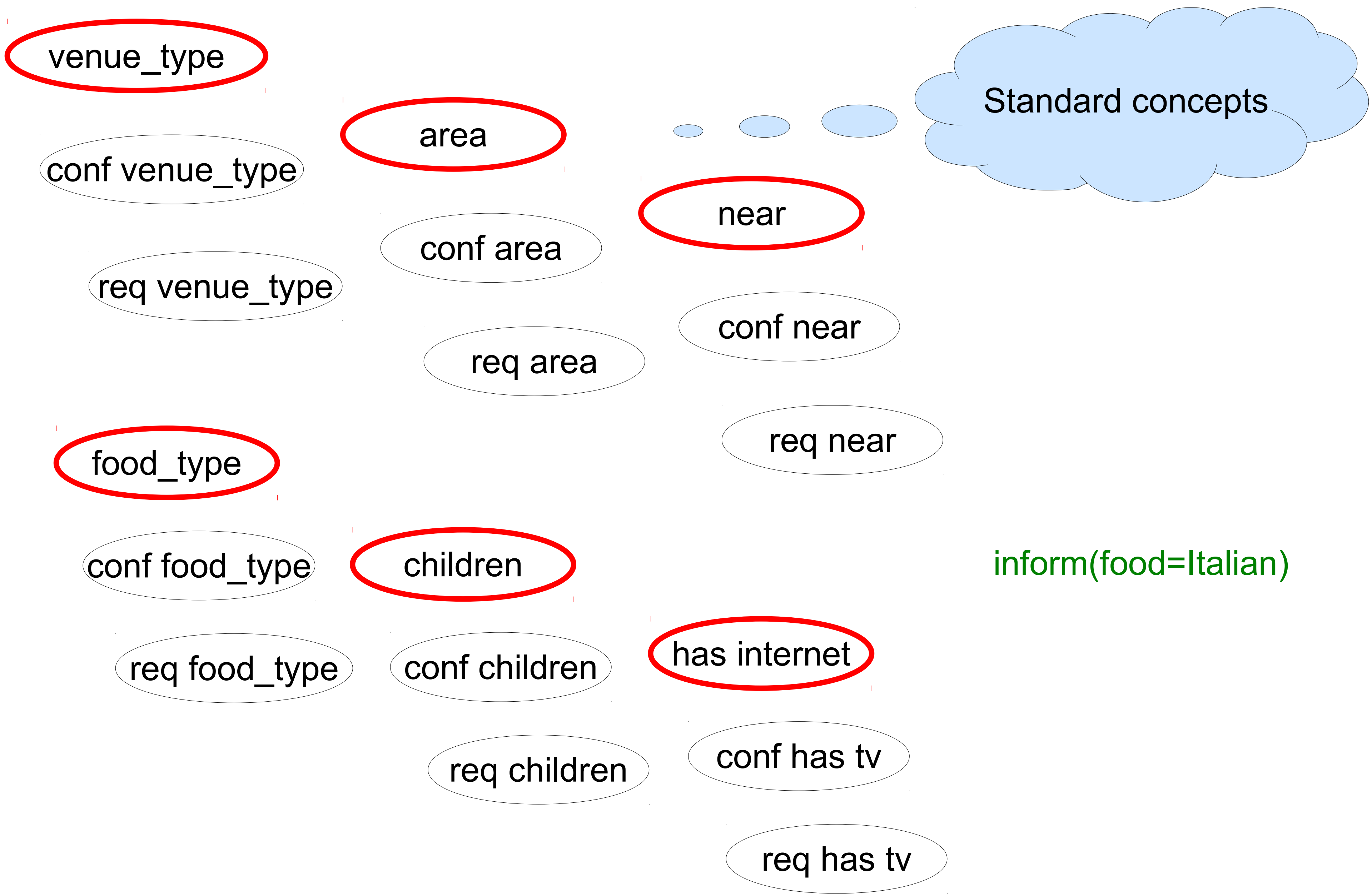


inform(food_type=Chinese)&inform(area=centre)

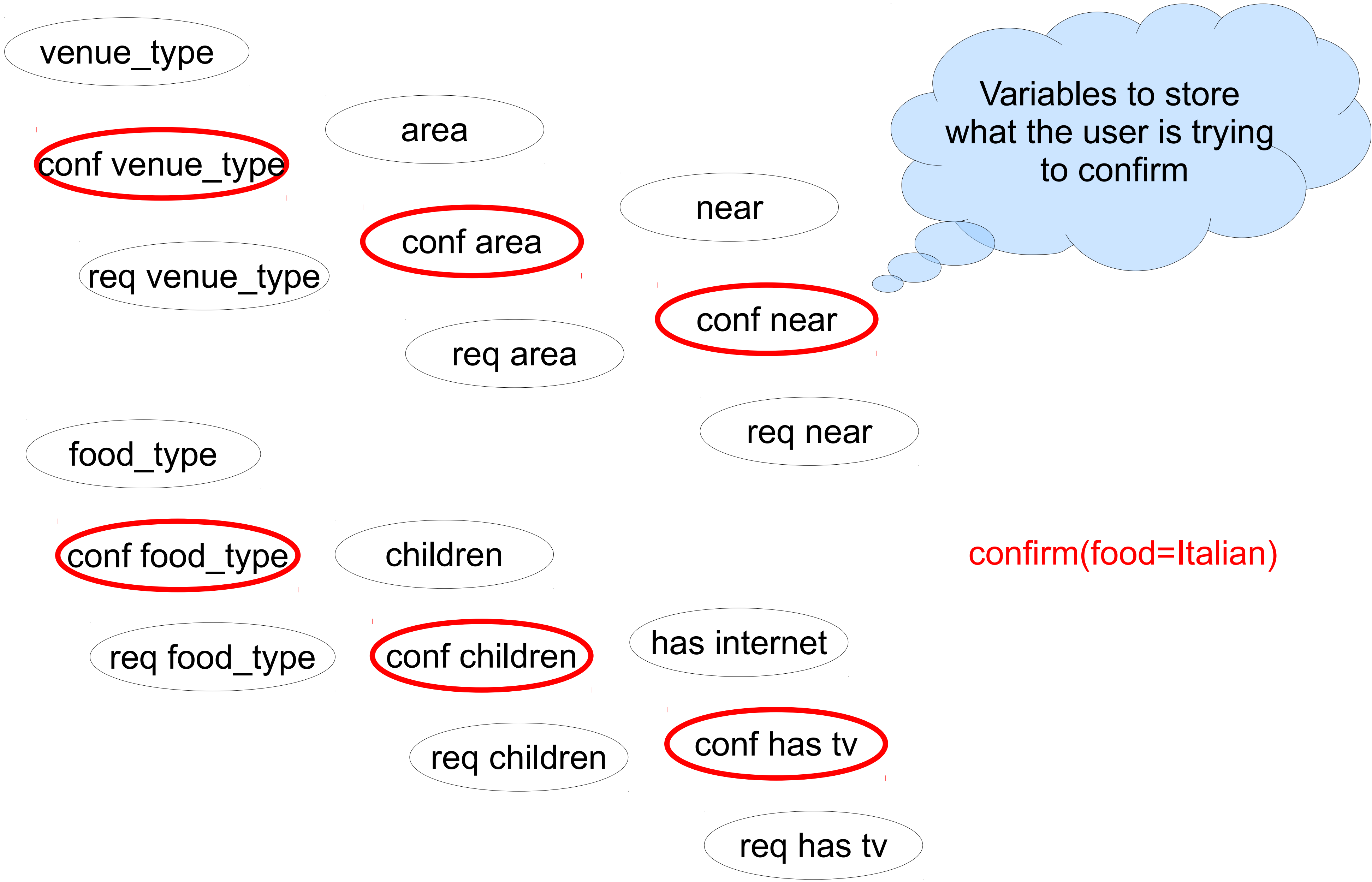
Dialogue state contains much more

- It should also store information about the context
- E.g.
  - what user requested
  - what user confirms
  - what system already informed about
  - what system confirmed

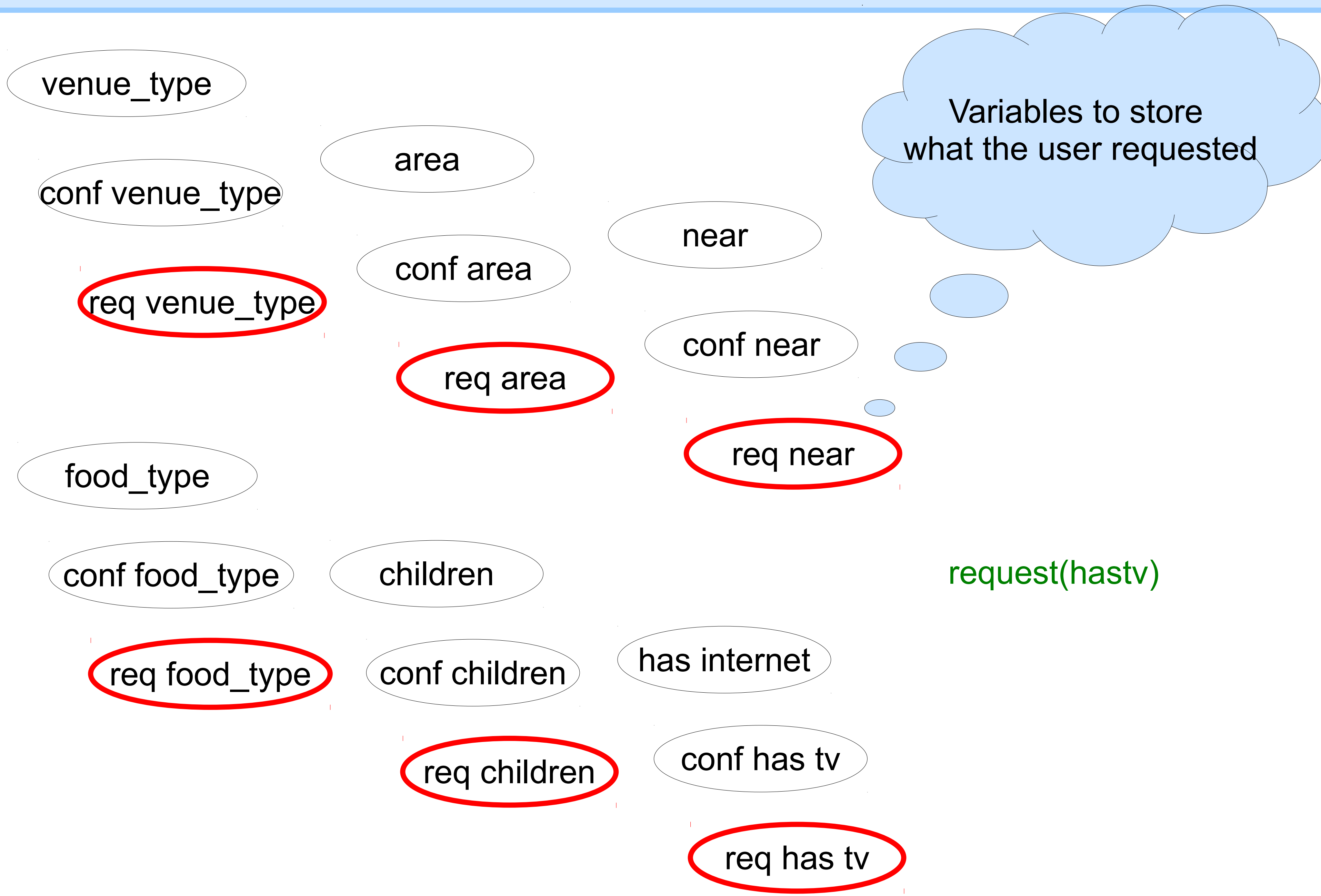
Expanded state



Expanded state



Expanded state



Expanded state

- Should also contain a variable for handling other semantic context
- “context node” possible values:
  - hello
  - bye
  - ack
  - thank you
  - request more
  - repeat
  - restart

Further expanded state

- Should store all info we gave to the user
- E.g.
  - names of the offered venues
  - the order of the offered venues
- To be able handle e.g. references
  - “No, I would the prefer the previous bar. Give me the address.”
- To offer an alternative
  - “Do you have anything else?”
  - give a user a venue which was not talked about yet

Summary so far

- How to define dialogue state
- Use of domain independent ontology
- The sate must support general aspects of a dialogue
  - such as negotiation

Problem with this approach

- ASR is unreliable
  - WER 30% in real-life environment
- SLU makes mistakes too
  - Some utterances are ambiguous
- Example:
  - User said:
    - I am looking for an inexpensive hotel.
  - ASR decoded:
    - I am looking for an expensive hotel.
  - SLU output:
    - inform(venue_type=hotel, price_range=expensive)

Example

- U1: inform(venue_type=restaurant)

- Dialogue state:
  - venue_type = restaurant
  - food_type = None
  - price_range = None
  - stars = None

Example

- U1: inform(venue_type=restaurant)
- U2: inform(stars=five)

Contradiction

- Dialogue state:
  - venue_type = restaurant
  - food_type = None
  - price_range = None
  - stars = five

Restaurants usually do not have stars

Attempts to fix the problem

- Detect contradictions
  - Confirm the contradicting information
- Reject input with low confidence score
  - inform(venue_type=restaurant,stars=five) [0.3]
- How to set the threshold?
- Are we loosing some information?
  - What if we reject something 10x?

Statistical Spoken Dialogue Systems

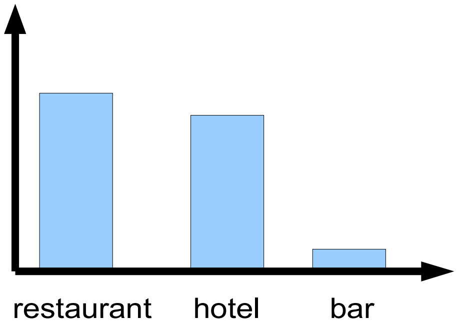
- Main goal is to make the dialogue systems
  - robust
  - natural

- Robustness
  - accumulation information over multiple turns
  - accumulating information from N-best list

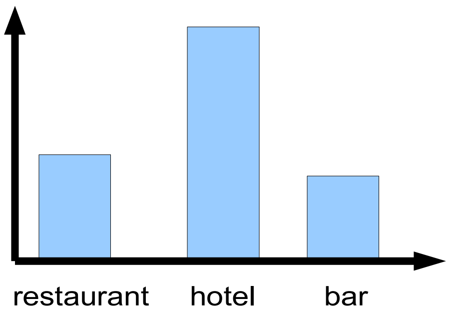
- Naturalness
  - trained from data / interaction with users

Information from multiple turns

- Accumulating the probabilities
- Turn 1:
  - inform(venue_type=restaurant) [0.5]
  - inform(venue_type=hotel) [0.4]

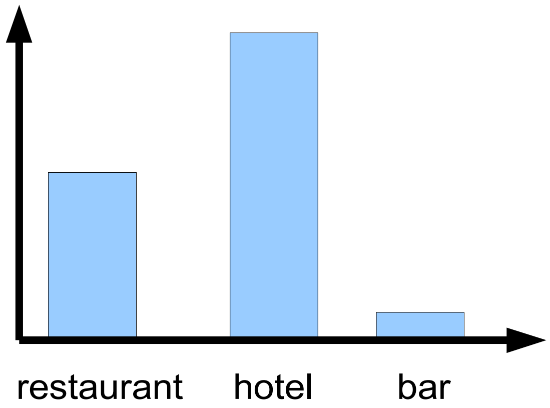


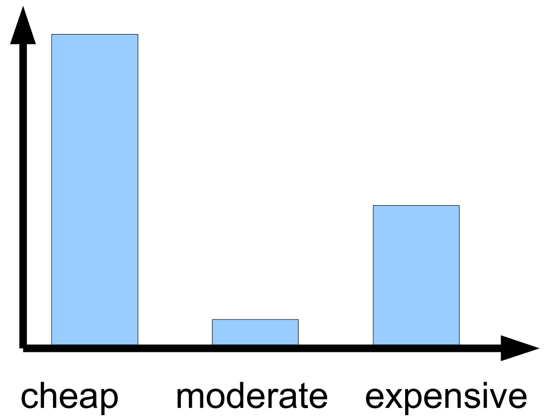
- Turn 2:
  - inform(venue_type=bar) [0.4]
  - inform(venue_type=hotel) [0.4]



Information from multiple hypotheses

- Accumulating the probabilities
- N-best list:
  - inform(venue_type=restaurant)&inform(price_range=cheap) [0.3]
  - inform(venue_type=hotel)&inform(price_range=expensive) [0.3]
  - inform(venue_type=hotel)&inform(price_range=cheap) [0.2]





POMDP motivation

- The previous behaviour can be elegantly handled by
  - Partially Observable Markov Decision Process (POMDP)

- Context can be used to resolve some ambiguity

- Context models can be optimised wit respect to the domain in hand

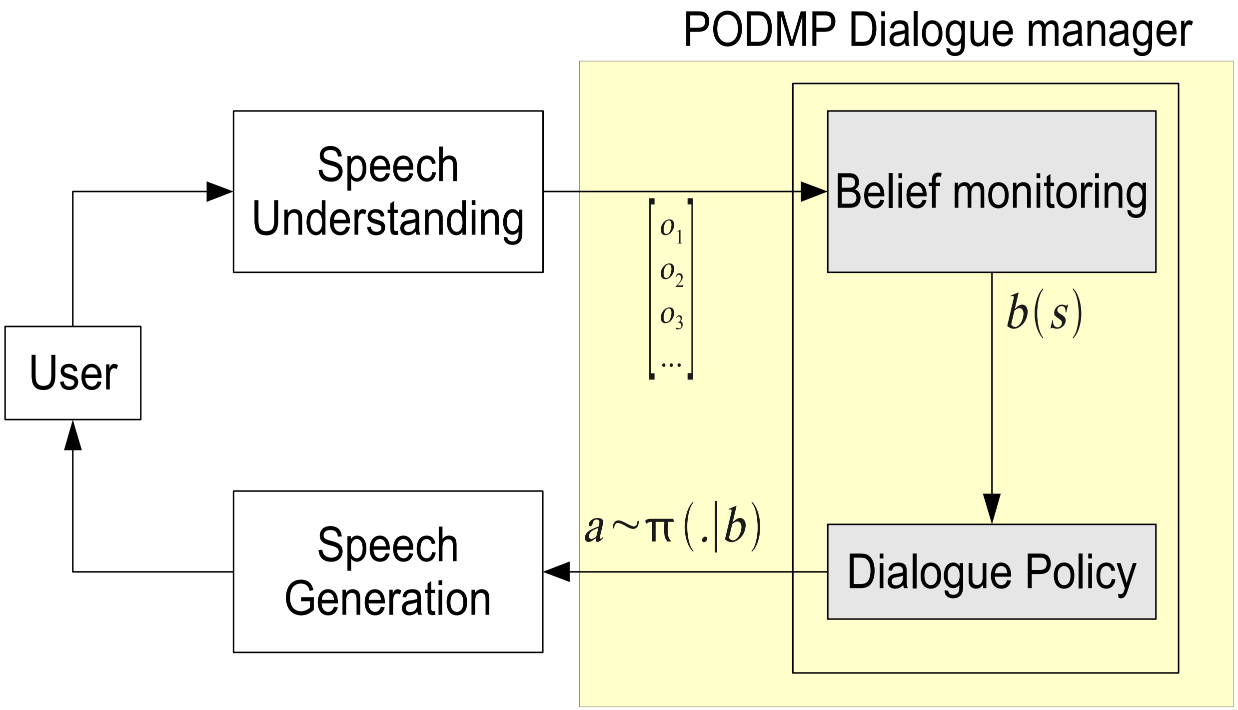
POMDP dialogue system #1

- POMDP DM can be naturally divided into two components:
  - belief monitoring
    - tracks what a user said – a distribution over all states
  - action selection
    - decides what to do next – a discrete action

We will talk about this later.

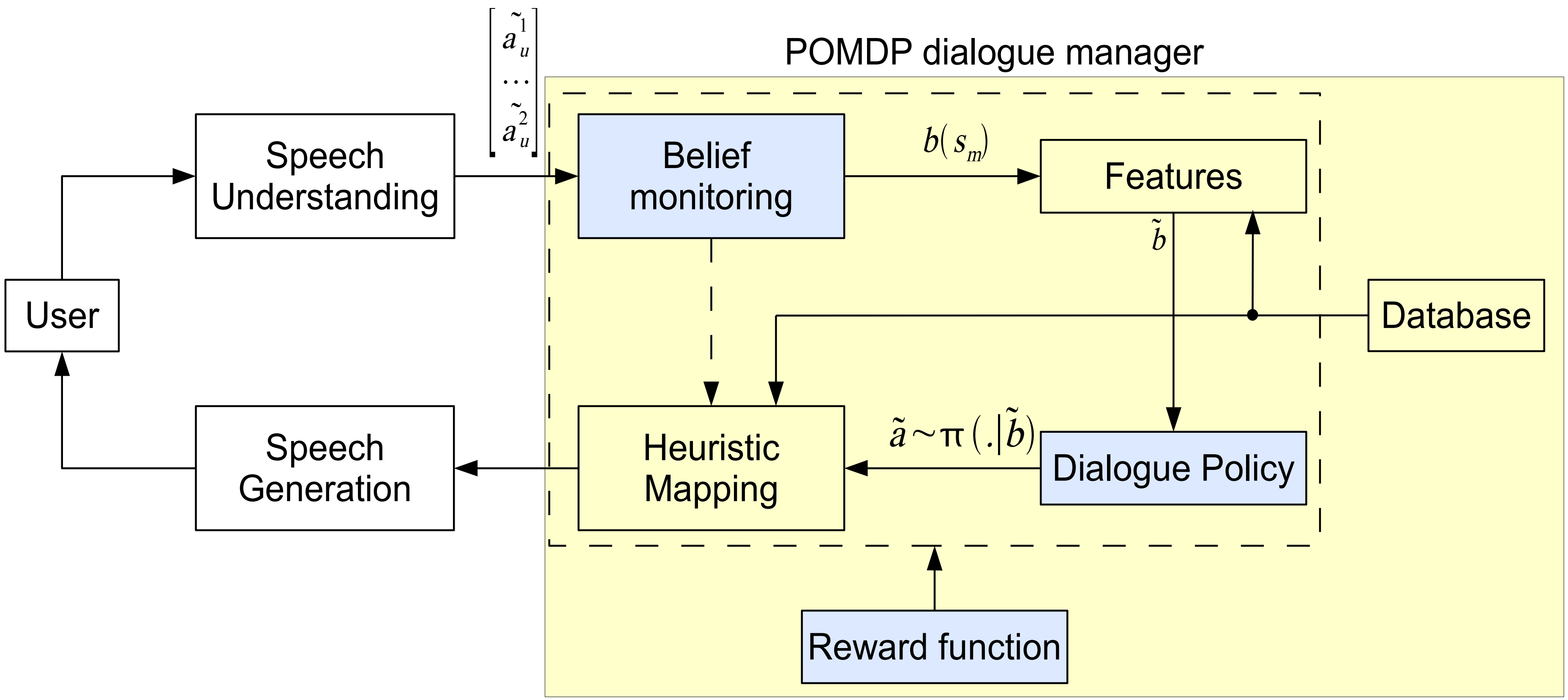
- When a POMDP system is trained using reinforcement learning then it is optimised to maximise a reward function
  - e.g. average success rate, length of a dialogue, both, etc.

POMDP dialogue system #2



- Instead of tracking the state ***s***, the dialogue manager maintains a distribution over all states: ***b(s)***
- Policy explicitly takes into the account in the uncertainty in ***b(s)***

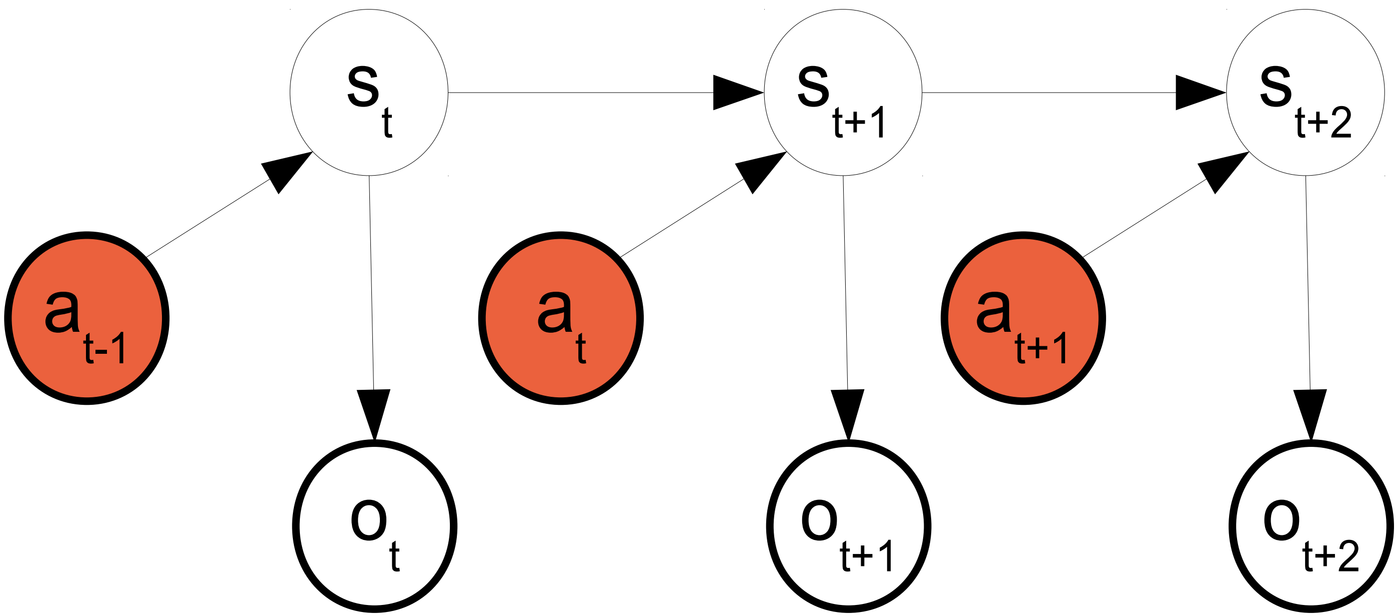
POMDP dialogue system approximations



Each of the components uses its own set of approximation techniques to achieve real-time performance

Belief monitoring

- Maintains prob. distribution over all possible states: **b(s)**
- Belief state **b(s)**
  - can be modelled as input-output HMM

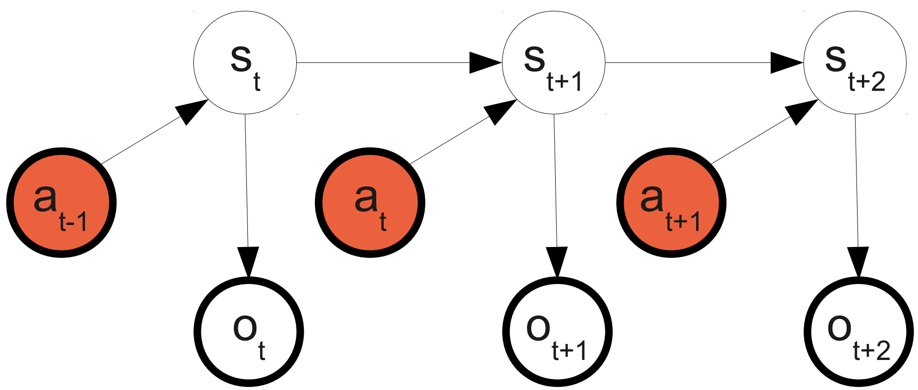


- a – the system's action - output
- o – user's actions - input

Naive belief monitoring

- The exact inference is trivial

$$b(s;\tau) \propto p(o_t|s_t;\tau) \sum_{s_{t-1}} p(s_t|a_{t-1}, s_{t-1};\tau)\, b(s_{t-1};\tau)$$



- The estimate depends on the dialogue model
  - parametrized by τ

- Problem is that there are too many states
  - e.g. 10 slots each with 10 values gives 10^{10} distinct states

Speeding things up

- Some researchers:
  - enumerate the most likely states and prune the others
  - mixture model belief monitoring
    - J. Henderson and O. Lemon, "**Mixture model POMDPs for efficient handling of uncertainty in dialogue management**," pp. 73-76, Jun. 2008.
  - group similar states
    - S. Young, M. Gasic, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson and K. Yu (2010). "**The Hidden Information State Model: a practical framework for POMDP-based spoken dialogue management.**"
  - particle filters
    - J. D. Williams, "**USING PARTICLE FILTERS TO TRACK DIALOGUE STATE**," in Proceedings of IEEE ASRU, 2007.
  - belief propagation
    - B. Thomson and S. Young (2010). "**Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems.**"

Enumerating and pruning

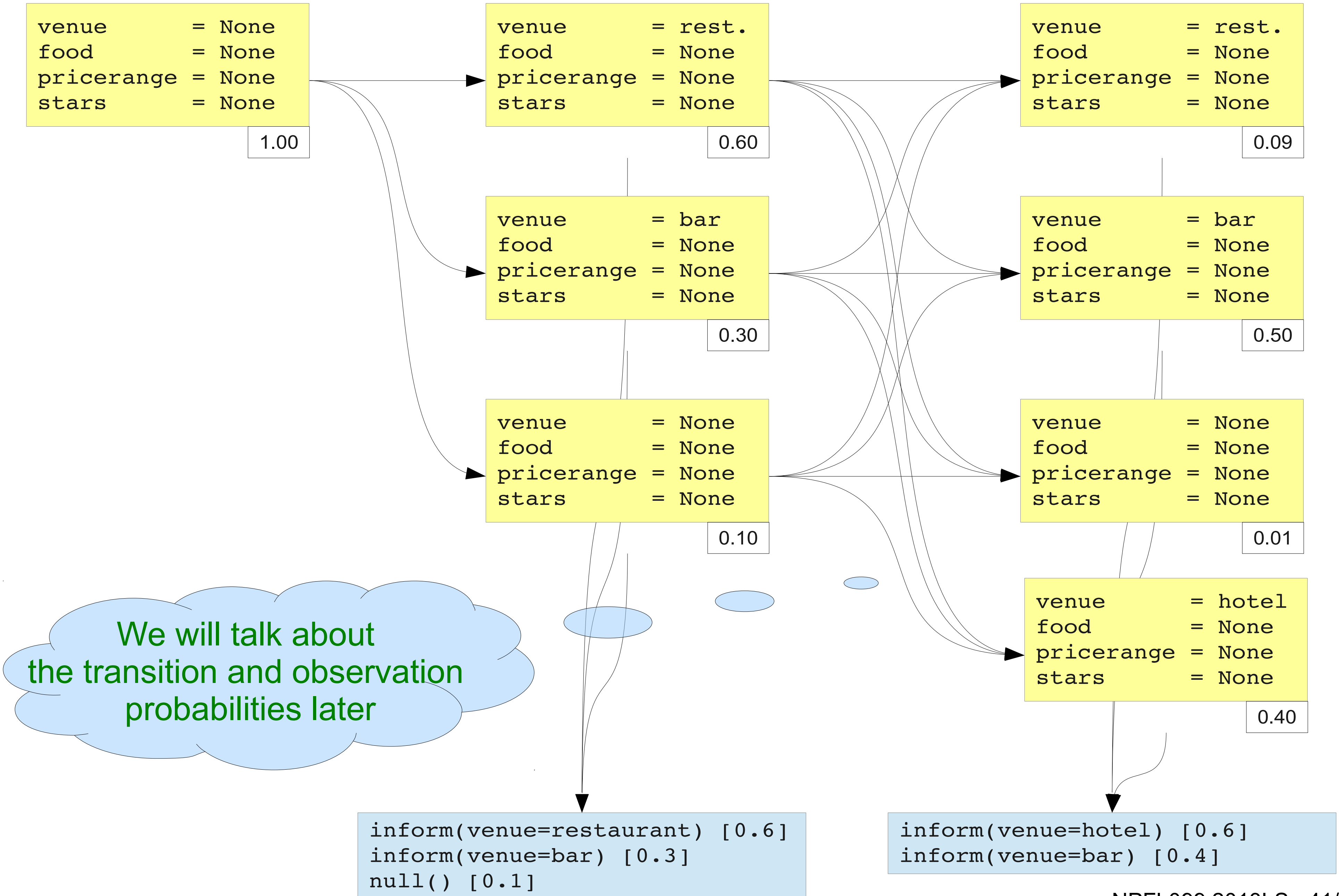
- Pruning less likely states
  - states with low probability are ignored
  - however, even after pruning, there are too many states

$$\begin{bmatrix} s_1 = 0.050 \\ s_2 = 0.100 \\ s_3 = 0.110 \\ s_4 = 0.300 \\ \cancel{s_5 = 0.001} \\ \ldots \end{bmatrix}$$

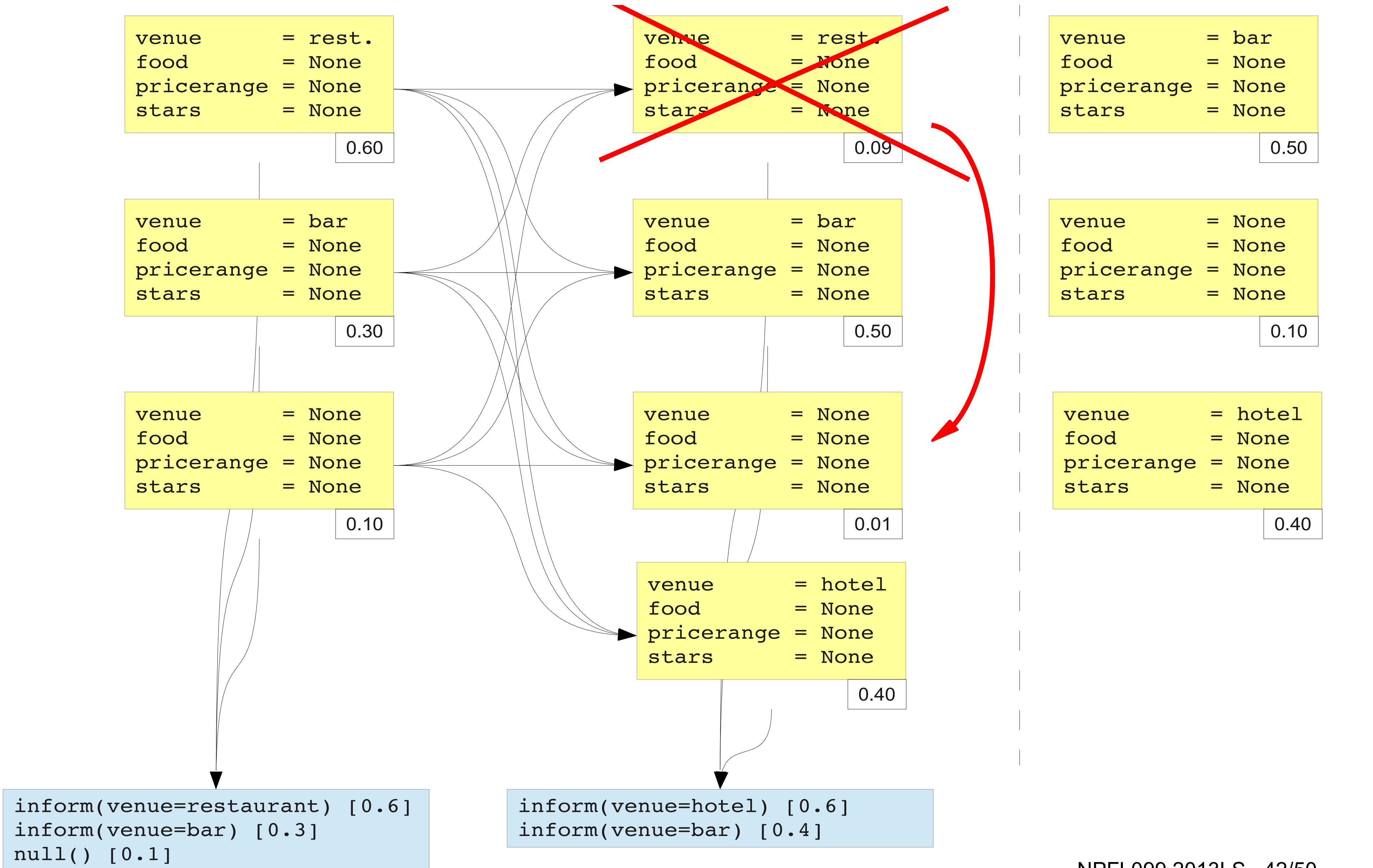
- Enumerate only states supported by observation hypotheses

  - T1:
    - inform(venue=restaurant) [0.6]
    - inform(venue=bar) [0.3]

Enumerating



Pruning


venue = rest.
food = None
pricerange = None
stars = None
0.60
venue = bar
food = None
pricerange = None
stars = None
0.30
venue = None
food = None
pricerange = None
stars = None
0.10
venue = rest.
food = None
pricerange = None
stars = None
0.09
venue = bar
food = None
pricerange = None
stars = None
0.50
venue = None
food = None
pricerange = None
stars = None
0.01
venue = hotel
food = None
pricerange = None
stars = None
0.40
venue = bar
food = None
pricerange = None
stars = None
0.50
venue = None
food = None
pricerange = None
stars = None
0.10
venue = hotel
food = None
pricerange = None
stars = None
0.40
inform(venue=restaurant) [0.6]
inform(venue=bar) [0.3]
null() [0.1]
inform(venue=hotel) [0.6]
inform(venue=bar) [0.4]

Dialogue model

- Model parameters can be estimated from some annotated data
  - very tedious

- Transition model: $p(s_{t+1} | s_t, a_t)$
  - models dynamics of the evolution of the sates
  - from a particular state to states generated based on the input observations/hypotheses

- Observation model: $p(o_t | s_t)$
  - models probability of the observations given a state

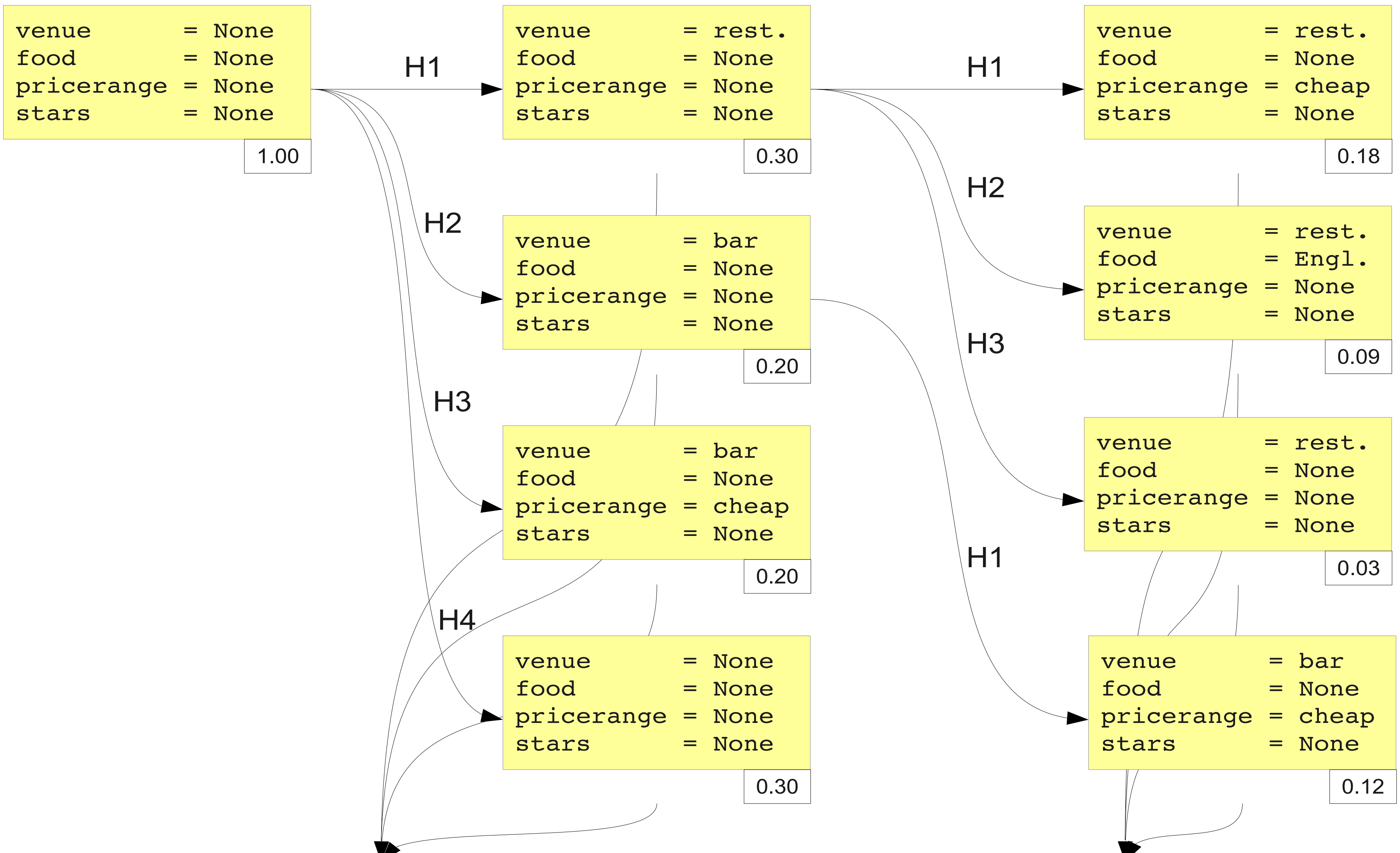
Mixture model belief monitoring

- Updating the dialogue state for each input hypothesis separately
- State probability depends only on observations
- Transitions allowed only between the “compatible states” given the observation

- Can be viewed as maintaining a set of dialogue managers executing in parallel

P. Crook, J. Henderson, O. Lemon, and X. Liu, “D1 . 3 : POMDP Learning for ISU Dialogue Management,” Learning, no. February, 2010.

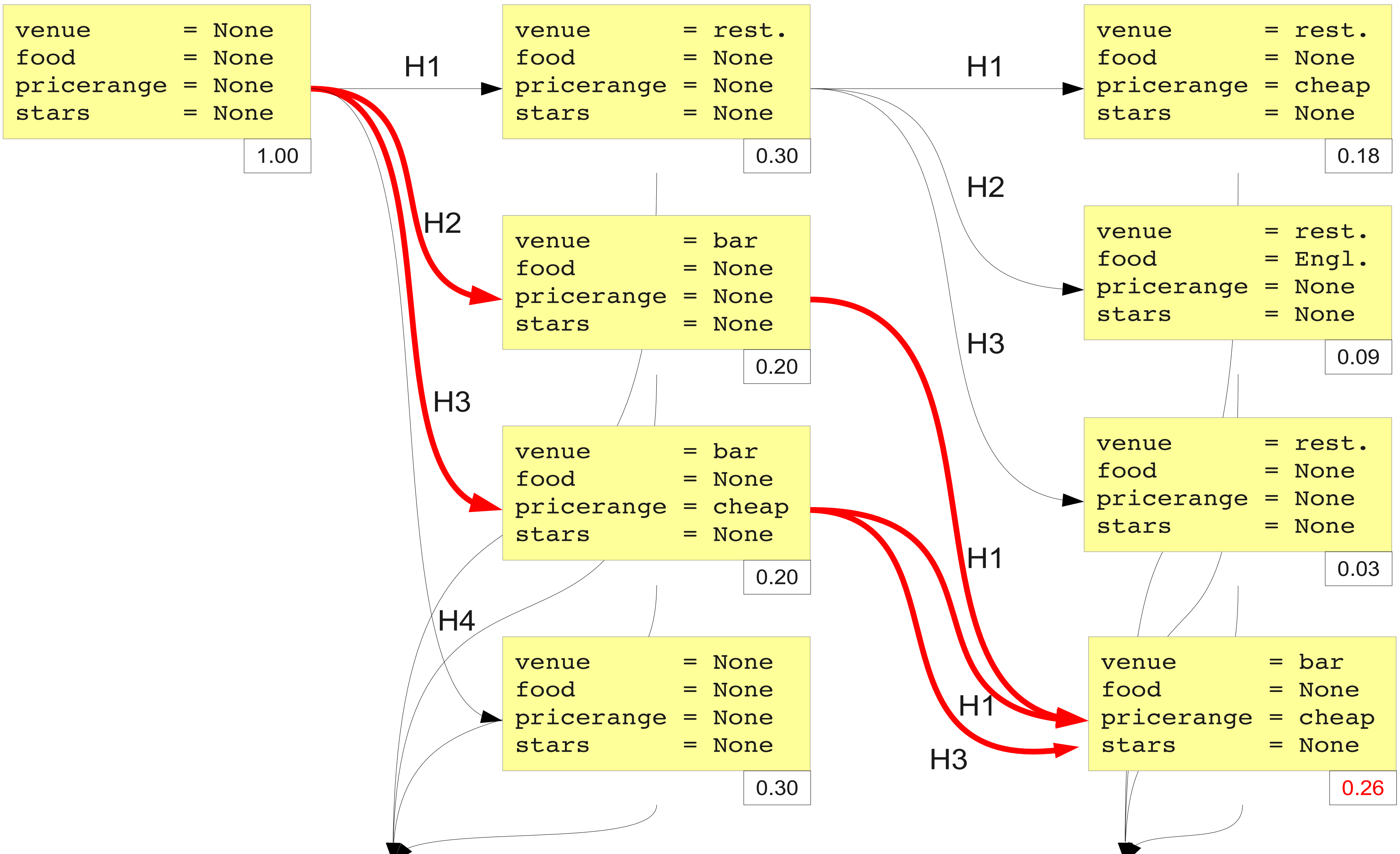
Enumerating




```
H1: inform(venue=restaurant) [0.3]
H2: inform(venue=bar) [0.2]
H3: inform(venue=bar)&inform(pricerange=cheap) [0.2]
H4: null() [0.3]
```

```
H1: inform(pricerange=cheap) [0.6]
H2: inform(food=English) [0.3]
H3: null() [0.1]
```

State merging




```
H1: inform(venue=restaurant) [0.3]
H2: inform(venue=bar) [0.2]
H3: inform(venue=bar)&inform(pricerange=cheap) [0.2]
H4: null() [0.3]
```

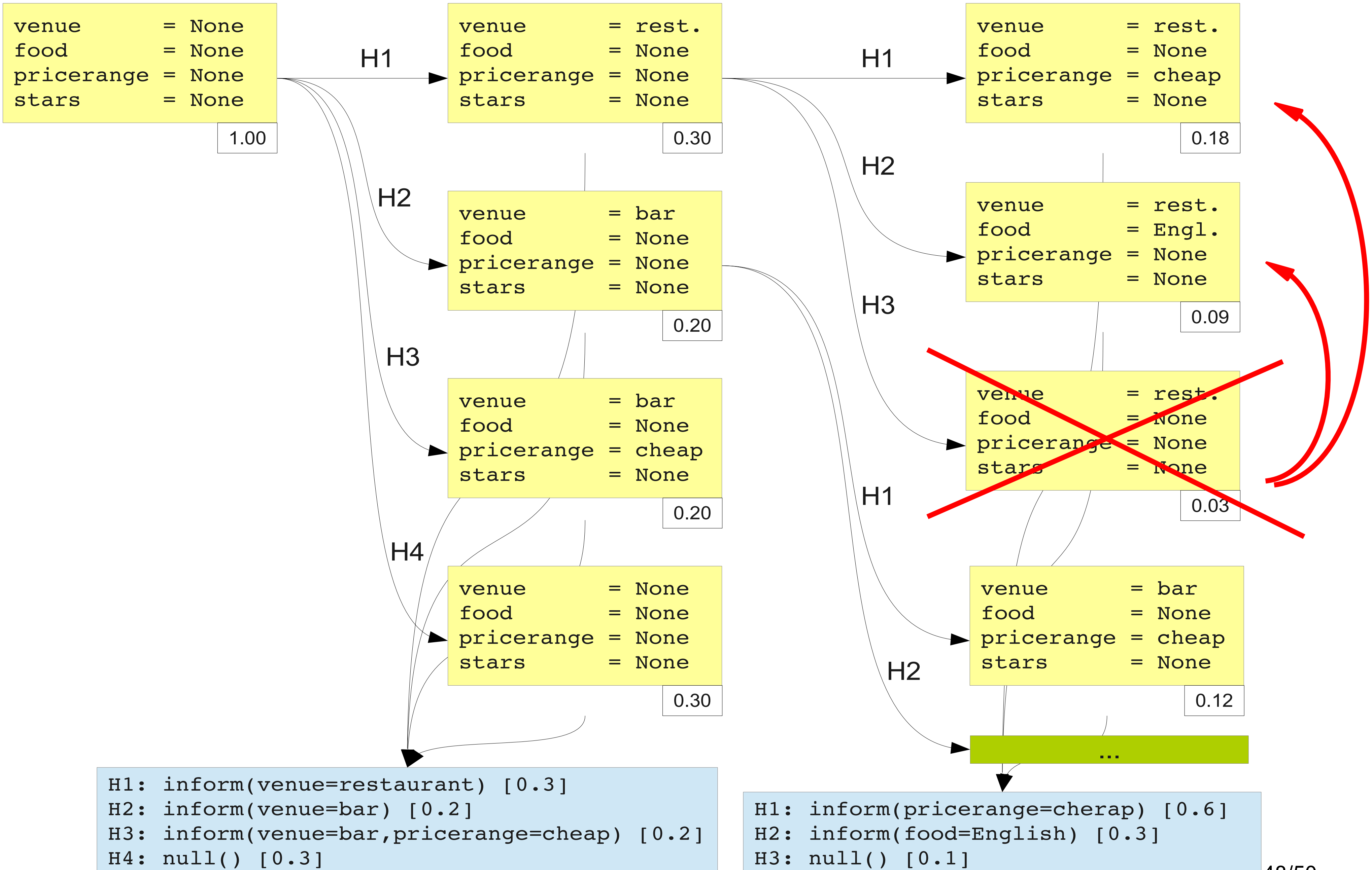
```
H1: inform(pricerange=cheap) [0.6]
H2: inform(food=English) [0.3]
H3: null() [0.1]
```

Pruning

- First
  - find similar states (e.g. share filled slots but not history, share some filled slots)
  - prune the less likely
  - add the pruned probability mass to the kept states
- Second
  - Prune states with low probability
  - Redistribute the probability mass
    - e.g. add the pruned probability mass to the initial state

- Pruning should not simply remove a hypothesis and renormalise, it should redistribute the probability of a pruned hypothesis to similar hypotheses

Pruning



venue = None
food = None
pricerange = None
stars = None
1.00
H1
H2
H3
H4
venue = rest.
food = None
pricerange = None
stars = None
0.30
venue = bar
food = None
pricerange = None
stars = None
0.20
venue = bar
food = None
pricerange = cheap
stars = None
0.20
venue = None
food = None
pricerange = None
stars = None
0.30
H1
H2
H3
H1
H2
venue = rest.
food = None
pricerange = cheap
stars = None
0.18
venue = rest.
food = Engl.
pricerange = None
stars = None
0.09
venue = rest.
food = None
pricerange = None
stars = None
0.03
venue = bar
food = None
pricerange = cheap
stars = None
0.12
...
H1: inform(venue=restaurant) [0.3]
H2: inform(venue=bar) [0.2]
H3: inform(venue=bar,pricerange=cheap) [0.2]
H4: null() [0.3]
H1: inform(pricerange=cherap) [0.6]
H2: inform(food=English) [0.3]
H3: null() [0.1]

Dialogue model

- Choice of the model can greatly simplify computation
- All prob. information comes only form the observation model

- Transition model:
  - $\sum_{s_{t+1} \in C(s_t, o_t)} p(s_{t+1} | s_t, a_t) = 1.0$
  - from a particular state to states generated based on the input observations/hypotheses
  - probability is uniform for all compatible states

- Observation model:
  - $p(o_t | s_t)$
  - the model is further factorised to prevent data sparsity

Thank you!

Filip Jurčíček

Institute of Formal and Applied Linguistics
Charles University in Prague
Czech Republic

Home page: http://ufal.mff.cuni.cz/~jurcicek