Probabilistic Ontology Trees for Belief Tracking in Dialog Systems

Neville Mehta
Oregon State University
mehtane@eecs.oregonstate.edu

Rakesh Gupta
Honda Research Institute
rgupta@hra.com

Antoine Raux
Honda Research Institute
araux@hra.com

Deepak Ramachandran
Honda Research Institute
dramachandran@hra.com

Stefan Krawczyk
Stanford University
stefank@cs.stanford.edu

Abstract

We introduce a novel approach for robust belief tracking of user intention within a spoken dialog system. The space of user intentions is modeled by a probabilistic extension of the underlying domain ontology called a probabilistic ontology tree (POT). POTs embody a principled approach to leverage the dependencies among domain concepts and incorporate corroborating or conflicting dialog observations in the form of interpreted user utterances across dialog turns. We tailor standard inference algorithms to the POT framework to efficiently compute the user intentions in terms of m-best most probable explanations. We empirically validate the efficacy of our POT and compare it to a hierarchical frame-based approach in experiments with users of a tourism information system.

1 Introduction

A central function of a spoken dialog system (SDS) is to estimate the user's intention based on the utterances. The information gathered across multiple turns needs to be combined and understood in context after automatic speech recognition (ASR). Traditionally, this has been addressed by dialog models and data structures such as forms (Goddeau et al., 1996) and hierarchical task decomposition (Rich and Sidner, 1998). To formalize knowledge representation within the SDS and enable the development of reusable software and resources, researchers have investigated the organization of domain concepts using IS-A/HAS-A ontologies (van Zanten, 1998; Noh et al., 2003).

Because the SDS only has access to noisy observations of what the user really uttered due to speech recognition and understanding errors, belief tracking in speech understanding has received particular attention from proponents of probabilistic approaches to dialog management (Bohus and Rudnicky, 2006; Williams, 2006). The mechanism for belief tracking often employs a Bayesian network (BN) that represents the joint probability space of concepts while leveraging conditional independences among them (Paek and Horvitz, 2000). Designing a domain-specific BN requires significant effort and expert knowledge that is not always readily available. Additionally, real-world systems typically yield large networks on which inference is intractable without major assumptions and approximations. A common workaround to mitigate the intensive computation of the joint distribution over user intentions is to assume full conditional independence between concepts which violates the ground truth in most domains (Bohus and Rudnicky, 2006; Williams, 2006).

We propose a novel approach to belief tracking for an SDS that solves both the design and tractability issues while making more realistic conditional independence assumptions. We represent the space of user intentions via a *probabilistic ontology tree* (POT) which is a tree-structured BN whose structure is directly derived from the hierarchical concept structure of the domain specified via an IS-A/HAS-A ontology. The specialization (IS-A) and composition (HAS-A) relationships between the domain concepts are intuitive and provide a systematic way of representing ontological knowledge for a wide range of domains.

The remainder of the paper is structured as follows. We begin by describing the construction of the POT given a domain ontology. We show how a POT employs null semantics to represent consistent user intentions based on the specialization and composition constraints of the domain. We then show how standard inference algorithms can be tailored to exploit the characteristics of the POT to efficiently infer the m-best list of probable explanations of user intentions given the observations. The POT and the associated inference algorithm empower a dialog manager (DM) to account for uncertainty while avoiding the design complexity, intractability issues, and other restrictive assumptions that characterize state-of-the-art systems. The section on empirical evaluation describes experiments in a tourist information domain that compare the performance of the POT system to a frame-based baseline system. The paper concludes with a discussion of related work.

2 Problem Formulation

Let $\{X_1, X_2, \ldots, X_N\}$ be a set of N concepts. Every concept X_i takes its value from its finite discrete domain $\mathcal{D}(X_i)$ which includes a special null element for the cases where X_i is irrelevant. The user intention space is defined as $\mathcal{U} = \mathcal{D}(X_1) \times \mathcal{D}(X_2) \times \cdots \times \mathcal{D}(X_N)$. At each dialog turn t, the system makes a noisy observation o_t about the true user intention $u \in \mathcal{U}$. o_t consists of a set of *slots*. A slot is a tuple $\langle v, d, c \rangle$ where $v \in \{X_1, \ldots, X_N\}$, $d \in \mathcal{D}(v)$ is a value of v, and $c \in \mathbb{R}$ is the confidence score assigned to that concept-value combination by the speech understanding (SU) system.

The *goal* of *belief tracking* is to maintain $\Pr(X_1, \ldots, X_N | o_1, \ldots, o_t)$, a distribution over the N-dimensional space $\mathcal{U}$ conditioned on all the observations made up to turn t. At each turn, the belief is updated based on the new observations to estimate the true, unobserved, user intention.

3 Probabilistic Ontology Trees

We model the space of the user intentions via a POT. A POT is a tree-structured BN that extends a domain ontology by specifying probability distributions over its possible instantiations based on specializations and compositions.

3.1 Domain Ontology

To ensure that the corresponding POTs are tree-structured, we consider a restricted class of domain ontologies over concepts.

Definition 1. *A* domain ontology *is a labeled directed acyclic graph. The set of vertices (corresponding to the domain concepts) is partitioned into $\{V_0\}$, V_S, and V_C, where V_0 is the only root node, V_S is the set of* specialization *nodes (related via* IS-A *to their parents), and V_C is the set of* composition *nodes (related via* HAS-A *to their parents). The set of edges satisfy the constraints that a specialization node has exactly one parent and a composition node may only have more than one parent if they are all specialization nodes with a common parent.*

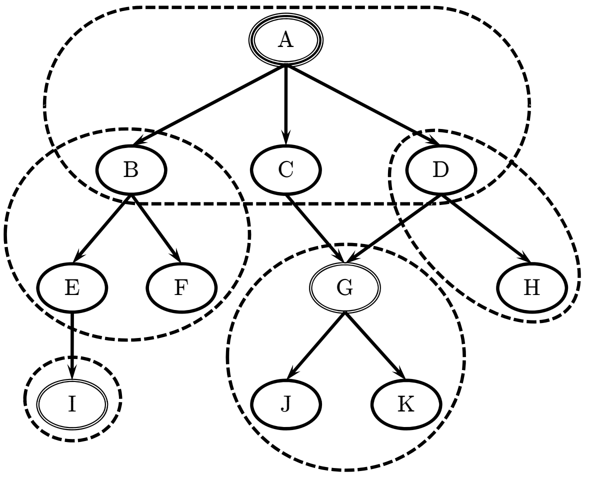


Figure 1: The ontology for a sample domain where B IS-A A, C IS-A A, D IS-A A, E IS-A B, F IS-A B, C HAS-A G (essential), D HAS-A G (nonessential), H IS-A D, E HAS-A I (essential), J IS-A G, and K IS-A G. Specialization nodes are drawn single-lined, composition nodes are drawn double-lined, and the root node is drawn triple-lined. Specialization subtrees are marked by dashed ovals.

Specialization nodes represent refinements of their parent concepts. Specializations of a concept are disjoint, that is, for any particular instance of the parent exactly one specialization is applicable and the rest are inapplicable. For example, if Dog IS-A Animal and Cat IS-A Animal, then Cat is inapplicable when Dog is applicable, and vice versa. Composition nodes represent attributes of their parents and may be essential or nonessential, e.g., Dog HAS-A Color (essential), Dog HAS-A Tail (nonessential). These definitions correspond with the standard semantics in the knowledge representation community (Noh et al., 2003). An example ontology is shown in Figure 1.

Definition 2. *A* specialization subtree (spec-tree) *in the ontology is a subtree consisting of a node with its specialization children (if any).*

3.2 POT Construction

We now describe how a POT may be constructed from a domain ontology. The purpose of the POT is to maintain a distribution of possible instantiations of the ontology such that the ontological structure is respected.

Given an ontology G, the corresponding POT is a tree-structured BN defined as follows:

Variables. Let T be a spec-tree in G with root R. Unless R is a (non-root) specialization node with no specialization children, T is represented in the POT by a variable X with the domain

$$\mathcal{D}(X) = \begin{cases} \{\mathsf{exists}, \mathsf{null}\}, \text{ if } \mathsf{Children}_T(R) = \varnothing \\ \mathsf{Children}_T(R), \text{ if } R = V_0 \\ \mathsf{Children}_T(R) \cup \{\mathsf{null}\}, \text{ otherwise.} \end{cases}$$

Edges. Let POT variables X and Y correspond to distinct spec-trees T_X and T_Y in G. There is a directed edge from X to Y if and only if either

- A leaf of T_X is the root of T_Y.
- There is an edge from a leaf in T_X to the non-specialization root of T_Y.
- There is an edge from the non-specialization root of T_X to that of T_Y.

Conditional Probability Tables (CPTs). If X (corresponding to spec-tree T_X) is the parent of Y (corresponding to spec-tree T_Y) in the POT, then Y's CPT is conditioned as follows:

- If T_Y is rooted at one of the leaves of T_X, then

  $$\Pr(Y = \mathsf{null} | X = \mathsf{Y}) = 0$$
  $$\Pr(Y = \mathsf{null} | X \neq \mathsf{Y}) = 1$$

  where Y is the domain value of X corresponding to child Y.
- If R is the root of T_X, and T_Y has a composition root node that is attached only to nodes in $S \subset \mathsf{Children}_{T_X}(R)$, then

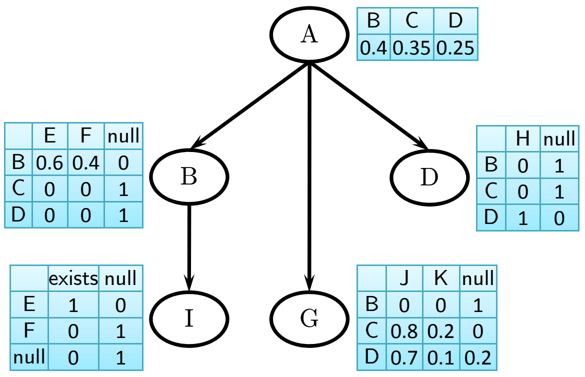
  $$\Pr(Y = \mathsf{null} | X = \mathsf{V}) = 1$$

  for any domain value V of X corresponding to a node $V \in \mathsf{Children}_{T_X}(R) - S$.
- If the root of T_Y is an essential composition node attached to a leaf V of T_X, then

  $$\Pr(Y = \mathsf{null} | X = \mathsf{V}) = 0$$

  where V is the domain value of X corresponding to the leaf V.

We label a POT variable with that of the root of the corresponding spec-tree for convenience. The domain of a POT variable representing a spec-tree comprises the specialization children (node names in sanserif font) and the special value null; the null value allows us to render any node (except the root) inapplicable. Spec-trees comprising single nodes have the domain value exists to switch between being applicable and inapplicable. The CPT entries determine the joint probabilities over possible valid instantiations of the ontology and could be based on expert knowledge or learned from data. The conditions we impose on them (*null semantics*) ensure that inconsistent instantiations of the ontology have probability 0 in the POT. While the ontology might have undirected cycles involving the children of spec-trees, the corresponding POT is a tree because spec-trees in the ontology collapse into single POT nodes. The POT for the example domain is shown in Figure 2.



B	C	D
0.4	0.35	0.25

	E	F	null
B	0.6	0.4	0
C	0	0	1
D	0	0	1

	H	null
B	0	1
C	0	1
D	1	0

	exists	null
E	1	0
F	0	1
null	0	1

	J	K	null
B	0	0	1
C	0.8	0.2	0
D	0.7	0.1	0.2

Figure 2: The POT for the example domain. If a node represents a spec-tree in the ontology, then it is labeled by the root of the spec-tree; otherwise, it is labeled with the name of the corresponding ontology node. $\mathcal{D}$(A) = {B, C, D}, $\mathcal{D}$(B) = {E, F, null}, $\mathcal{D}$(D) = {H, null}, and Pr(A), Pr(B|A) and Pr(D|A) represent some distributions over the respective specializations. $\mathcal{D}$(I) = {exists, null} and $\mathcal{D}$(G) = {J, K, null}. Note that a composition node (G) can be shared between multiple specializations (C and D) in the ontology while the resulting POT remains tree-structured.

3.3 Tourist Information POT

For the empirical analysis, we designed a POT for a tourist information system that informs the user about places to shop, eat, get service, and displays relevant information such as the distance to an intended location. The user can also provide conversational commands such as stop, reset, undo, etc. The full ontology for the tourist information domain is shown in Figure 3 and the POT is in Figure 4. In the POT, Action is the root node, with $\mathcal{D}$(Action) = {Venue, Command}, and $\mathcal{D}$(Venue)

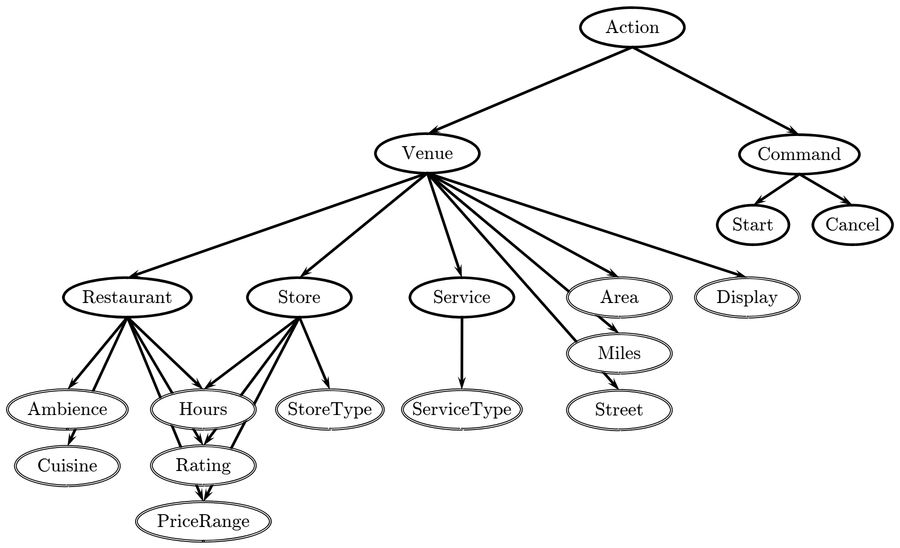


Figure 3: The ontology for the tourist information domain. All the composition nodes have specializations of their own (such as Japanese and Greek for Cuisine), but have not been shown for the sake of compactness.

= {Restaurant, Store, Service, null}. All the composition (or attribute) nodes such as Hours and Rating are made children of Venue by construction. Since a Command is inapplicable when the Action is a Venue, we have $\Pr$(Command = null | Action = Venue) = 1. The composition nodes (Cuisine, Street, etc.) have specializations of their own ({Japanese, Greek, ...}, {Castro, Elm, ...}, etc.), but are not shown for the sake of clarity. Since Cuisine is an essential attribute of Restaurant, $\Pr$(Cuisine = null | Venue = Restaurant) = 0; moreover, $\Pr$(Cuisine = null | Venue = Service) = 1 because Cuisine is not relevant for Service.

4 Inferring User Intention

We have seen how the POT provides the probabilistic machinery to represent domain knowledge. We now discuss how the POT structure can be leveraged to infer user intention based on the slots provided by the SU.

4.1 Soft Evidence

Every slot retrieved from the SU needs to be incorporated as observed evidence in the POT. We can set the associated node within the POT directly to its domain value as hard evidence when we know these values with certainty. Instead, we employ probabilistic observations to soften the evidence entered into the POT. We assume that the confidence score $c \in [0, 100]$ of a slot corresponds to the degree of certainty in the observation. For an observed slot variable X, we create an observation node $\hat{X}$ on the fly with the same domain as X and make it a child of X. If x is the observed value for slot X, then the CPT of $\hat{X}$ is constructed from the slot's confidence score as follows:

$$\Pr(\hat{X}|X = \mathsf{x}) = \begin{cases} \frac{c(|\mathcal{D}(X)|-1)/100+1}{|\mathcal{D}(X)|}, & \hat{X} = \mathsf{x} \\ \frac{1-c/100}{|\mathcal{D}(X)|}, & \hat{X} \neq \mathsf{x} \end{cases}$$

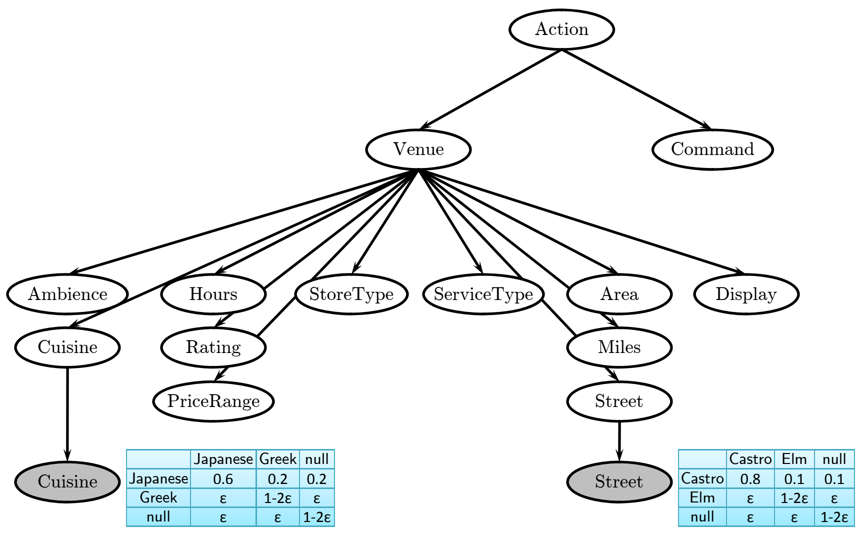
The probability values are generated by linearly interpolating between the uniform probability value and 1 based on the confidence score. For the remaining values,

$$\Pr(\hat{X}|X \neq \mathsf{x}) = \begin{cases} 1 - \varepsilon(|\mathcal{D}(X)| - 1), & \hat{X} = X \\ \varepsilon, & \hat{X} \neq X \end{cases}$$

where $\varepsilon > 0$.[1] Since the confidence score gives an indication of the probability for the observed value of a slot but says nothing about the remaining values, the diagonal elements for the remaining values are near 1. We cannot make them exactly 1 because the observation node needs to coexist with possibly conflicting observations in the POT.

If the user confirms the current POT hypothesis, then observations corresponding to the current hypothesis (with CPTs proportional to the score of the confirmation) are added to the POT to enforce the belief. If the user denies the current hypothesis, then all observations corresponding to the current hypothesis are removed from the POT.

[1]In our experiments, we use $\varepsilon = 10^{-10}$.



	Japanese	Greek	null
Japanese	0.6	0.2	0.2
Greek	ε	1-2ε	ε
null	ε	ε	1-2ε

	Castro	Elm	null
Castro	0.8	0.1	0.1
Elm	ε	1-2ε	ε
null	ε	ε	1-2ε

Figure 4: The POT for the tourist information domain. Assuming that $\mathcal{D}$(Cuisine) = {Japanese, Greek, null} and $\mathcal{D}$(Street) = {Castro, Elm, null}, the shaded observation nodes represent the soft evidence for input slots ⟨Cuisine, Japanese, 40⟩ and ⟨Street, Castro, 70⟩.

The POT for the tourist information domain after getting two slots as input is shown in Figure 4. The attached nodes are set to the observed slot values and the evidence propagates through the POT as explained in the next section.

4.2 POT Inference

A probable explanation (PE) or hypothesis is an assignment of values to the variables in the POT, and the most probable explanation (MPE) within the POT is the explanation that maximizes the joint probability conditioned on the observed variables. The top m estimates of the user's intentions correspond to the m-best MPEs. The design of the POT ensures that the m-best MPEs are all *consistent* across specializations, that is, exactly one specialization is applicable per node in any PE; all inconsistent explanations have a probability of 0.

The m-best MPEs could be found naively using the Join-Tree algorithm to compute the joint distribution over all variables and then use that to find the top m explanations. The space required to store the joint distribution alone is $O(n^N)$, where N is the number of nodes and n the number of values per node. Because the run time complexity is at least as much as this, it is impractical for any reasonably sized tree. However, we can get a significant speedup for a fixed m by using the properties of the POT.

Algorithm 1 uses a message-passing protocol, similar to many in the graphical models literature (Koller and Friedman, 2009), to simulate a dynamic programming procedure across the levels of the tree (see Figure 5). In Algorithm 2, an MPE message is computed at each node X using messages from the children, and sent to the parent. The message from X is the function (or table) $\psi_X(x, \vec{z})$ that represents the probabilities of the top m explanations, $\vec{z}$, of the subtree rooted at X for a particular value of $X = x$. At the root node X_0 we try all values of x_0 to find the top m MPEs for the entire tree. Note that in step 7, we

Algorithm 1 COMPUTE-PE
Input: POT T with root X_0, number of MPEs m, evidence E
Output: m MPEs for T

```
1: for $X \in T$ in reverse topological order do
2:     Collect messages $\psi_{Y_i}$ from all children $Y_i$ of $X$
3:     $\psi_X$ = COMPUTE-MPE-MESSAGE$(X, m, \{\psi_{Y_i}\})$
4: end for
5: return top $m$ elements of $\Pr(X_0|E)\psi_{X_0}(\cdot)$ without $E$
```

Algorithm 2 COMPUTE-MPE-MESSAGE
Input: POT node X, number of MPEs m, messages from children ψ_{Y_i}
Output: Message $\psi_X(\cdot)$

```
 1: if $X$ is a leaf node then
 2:     $\psi_X(x) \leftarrow 1, \forall x \in \mathcal{D}(X)$
 3:     return $\psi_X$
 4: end if
 5: for $x \in \mathcal{D}(X)$ do
 6:     for $\vec{z} = ((y_1, \vec{z_1}), \ldots, (y_k, \vec{z_k})) \in \{\mathcal{D}(\psi_{Y_1}) \times \ldots \times$
        $\mathcal{D}(\psi_{Y_k}) : \Pr(Y_i = \text{null}|X = x, E) < 1\}$ do
 7:         $\psi'_X(x, \vec{z}) \leftarrow \prod_i [\Pr(Y_i = y_i|X = x, E)\psi_{Y_i}(y_i, \vec{z_i})]$
 8:     end for
 9:     $\psi_X(x) \leftarrow$ top $m$ elements of $\psi'_X(x)$.
10: end for
11: return $\psi_X$
```

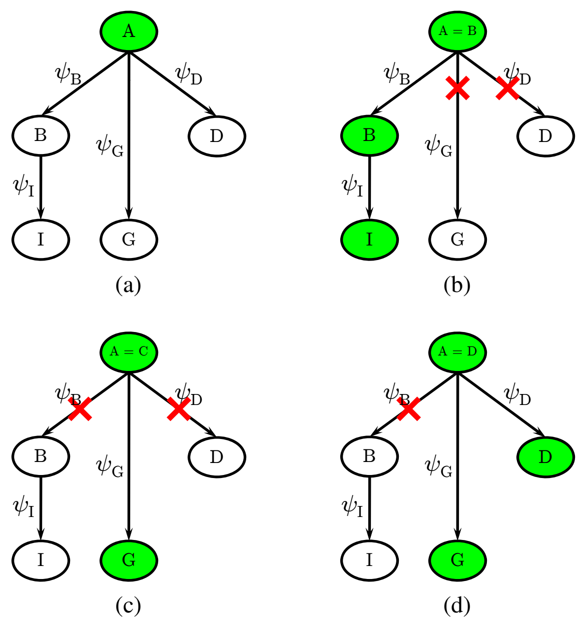


Figure 5: COMPUTE-MPE applied to the example POT. (a) Inference starts with the messages being passed up from the leaves to the root A. Every message ψ_X is an $m \times n$ table that contains the probabilities for the m-best MPEs of the subtree rooted at X for all the n domain values of X. (b) At the root, A is set to its first element B, and its marginal $\Pr(A = B)$ is combined with the message ψ_B. The semantics of the POT ensures that the other messages can be safely ignored because those subtrees are known to be null with probability 1. (c) A is set to C and only the essential attribute G is non-null. (d) A is set to its final element D, and consequently both the node D and the nonessential attribute G are non-null and their messages are mutually independent.

need the marginal $P(Y|X, E)$ which can be efficiently computed by a parallel message-passing method. Evidence nodes can only appear as leaves because of our soft evidence representation, and are encompassed by the base case. The algorithm leverages the fact that the joint of any entire subtree rooted at a node that is null with probability 1 can be safely assumed to be null with probability 1. The validity of Algorithm 1 is proven in Appendix A.

4.3 Complexity Analysis

At a POT node with at most n values and branching factor k, we do n maximizations over the product space of k nm-sized lists. Thus, the time complexity of Algorithm 1 on a POT with N nodes is $O(N(nm)^k)$ and the space complexity is $O(Nnmk)$. (Insertion sort maintains a sorted list truncated at m elements to keep track of the top m elements at any time.) However, the algorithm is significantly faster on specialization nodes because only one child is applicable and needs to be considered in the maximization (step 7). In the extreme case of a specialization-only POT, the time and space complexities both drop to $O(Nmn)$.

A similar algorithm for incrementally finding m-best MPEs in a general BN is given in Srinivas and Nayak (1996). However, our approach has the ability to leverage the null semantics in POTs resulting in significant speedup as described above. This is crucial because the run-time complexity of enumerating MPEs is known to be P^{PP}-Complete for a general BN (Kwisthout, 2008).

5 Empirical Evaluation

To test the effectiveness of our POT approach, we compare it to a frame-based baseline system for inferring user intentions.

The baseline system uses a hierarchical frame-based approach. Each frame maps to a particular user intention, and the frames are filled concurrently from the dialog observations. The slots from a turn overwrite matching slots received in previous turns. The baseline system uses the same ontology as the POT to insure that it only produces consistent hypotheses, e.g., it never produces ”Venue=Service, Cuisine=Japanese” because Service does not have a Cuisine attribute. When several hypotheses compete, the system selects the one with the maximum allocated slots. We implemented the POT engine based on the Probabilistic Network Library (Intel, 2005). It takes a POT specification as input, receives the ASR slots, and returns its m-best MPEs.

Using a tourism information spoken dialog system, we collected a corpus of 375 dialogs from 15 users with a total of 720 turns (details in Appendix B). Evaluation is performed by running these collected dialogs in batch and providing the ASR slots of each turn to both the baseline and POT belief-tracking systems.[2] After each turn, both systems return their best hypothesis of the overall user intention in the form of a set of concept-value pairs. These hypothe-

[2]Speech recognition and understanding was performed using the Nuance Speech Recognition System v8.5 running manual and statistical grammars with robust interpretation.

System		Precision	Recall	F1
POT	Top hypothesis	0.84	0.81	0.83
	Top 2 hypotheses	0.87	0.84	0.85
	Top 3 hypotheses	0.89	0.85	0.87
	Top 4 hypotheses	0.91	0.86	0.89
	Top 5 hypotheses	0.92	0.86	0.89
Baseline		0.84	0.79	0.81

Table 1: Precision/recall results comparing the baseline system against the POT-based system on the 25-scenario experiment. Results are averaged over all 15 users.

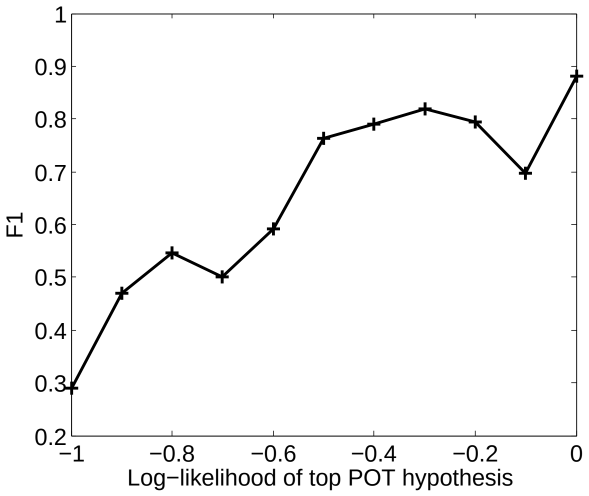


Figure 6: F1 score as a function of the log-likelihood of the top hypothesis for the user's goal.

ses are compared to the true user intention expressed so far in the dialog (e.g., if the user wants a cheap restaurant but has not mentioned it yet, PriceRange=Cheap is not considered part of the ground truth). This offline approach allows us to compare both versions on the same input.

Table 1 shows the precision/recall results for the experiment based on comparing the set of true user intention concepts to the inferred hypotheses of the POT and baseline systems. The average word error rate for all users is 29.6%. The POT system shows a 2% improvement in recall and F1 over the baseline. Additionally, leveraging the m-best hypotheses beyond just the top one could help enhance performance or guide useful clarification questions as shown by the improved performance when using the top 2–5 hypotheses; we assume an oracle for selecting the hypothesis with highest F1 among the top *m* hypotheses. All of the CPTs in the POT (besides the structural constraints) are uniformly distributed. Thus, the performance of the POT could be further improved by training the CPTs on real data.

To assess the quality of likelihood returned by the POT as a belief confidence measure, we binned dialog turns according to the log-likelihood of the top hypothesis and then computed the F1 score of each bin. Figure 6 shows that belief log-likelihood is indeed a good predictor of the F1 score. This information could be very useful to a dialog manager to trigger confirmation or clarification questions for example.

6 Discussion

The definition and construction of POTs provide a principled and systematic way to construct probabilistic models for an SDS. While any BN can be used to model the space of user intentions, designing an effective network is not an easy task for system designers not well versed in graphical models. In previous belief tracking work, researchers describe their networks with little indication on how they arrived at the specific structure (Paek and Horvitz, 2000; Thomson and Young, 2009). Prior work on ontologies for SDSs (van Zanten, 1998; Noh et al., 2003) as well as the prominence of concept hierarchies in other areas such as object-oriented programming and knowledge engineering make them a natural and intuitive way of representing SDS domains. The development of POTs builds on past research on constructing BNs based on ontological knowledge (Helsper and van der Gaag, 2002; Pfeffer et al., 1999).

While most approaches to belief tracking in the dialog systems community make a strict independence assumption between concepts (Bohus and Rudnicky, 2006; Williams, 2006), POTs model the dependencies between concepts connected by specialization and composition relationships while remaining significantly more tractable than general BNs and being very straightforward to design. The null semantics allow a POT to capture disjoint values and the applicability of attributes which are common aspects of concept ontologies. Obviously, a POT cannot capture all types of concept relationships since each concept can have only one parent. However, this restriction allows us to perform efficient exact computation of the m-best MPEs which is a significant advantage. Statistical Relational Learning approaches such as Markov Logic Networks (Richardson and Domingos, 2006) have been developed for more general relational models than strict ontologies, but they lack the parsimony and efficiency of POTs.

Thomson and Young (2009) describe an approach to dialog management based on a partially observable Markov decision process (POMDP) whose policy depends only on individual concepts' marginal distributions rather than on the overall user intention. Because their system performs belief tracking with a dynamic Bayesian network (DBN) rather than a static BN, the exact marginal computation is intractable and the authors use loopy belief propagation to compute the marginals. Even then, they indicate that the dependencies of the subgoals must be limited to enable tractability. In practice, all concepts are made independent except for the binary validity nodes that deterministically govern the dependence between nodes (similar to the null semantics of a POT). Williams (2007) also represents the user goal as a DBN for a POMDP-based DM. They perform belief updating using particle filtering and approximate the joint probability over the user intention with the product of the concept marginals. This could lead to inaccurate estimation for conditionally dependent concepts.

Among authors who have used m-best lists of dialog states for dialog management, Higashinaka et al. (2003) have shown empirically that maintaining multiple state hypotheses facilitates shorter dialogs. Their system scores each dialog state using a linear combination of linguistic and discourse features, and this score is used by a handcrafted dialog policy. While illustrating the advantages of m-best lists, this scoring approach lacks theoretical justification and ability to include prior knowledge that POTs inherit from BNs.

7 Conclusion

We have presented the POT framework for belief tracking in an SDS. We have shown how a POT can be constructed from the domain ontology and provided an exact algorithm to infer the user's intention in real-time. POTs strike a balance between representing rich concept dependencies and facilitating efficient tracking of the m-best user intentions based on exact joint probabilities rather than approximations such as concept marginals.

References

D. Bohus and A. Rudnicky. 2006. A K Hypotheses + Other Belief Updating Model. In *AAAI Workshop on Statistical and Empirical Approaches to Spoken Dialogue Systems*.

D. Goddeau, H. Meng, J. Polifroni, S. Seneff, and S. Busayapongchai. 1996. A Form-Based Dialogue Manager for Spoken Language Applications. In *ICSLP*.

E. Helsper and L. van der Gaag. 2002. Building Bayesian Networks through Ontologies. In *European Conference on Artificial Intelligence*.

R. Higashinaka, M. Nakano, and K. Aikawa. 2003. Corpus based Discourse Understanding on Spoken Dialog Systems. In *Annual Meeting on Association for Computational Linguistics*.

Intel. 2005. Probabilistic Network Library. `http://sourceforge.net/projects/openpnl/`.

D. Koller and N. Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

J. Kwisthout. 2008. Complexity Results for Enumerating MPE and Partial MAP. In *European Workshop on Probabilistic Graphical Models*.

H. Noh, C. Lee, and G. Lee. 2003. Ontology-based Inference for Information-seeking in Natural Language Dialog Systems. In *IEEE International Conference on Industrial Informatics*.

T. Paek and E. Horvitz. 2000. Conversation as Action under Uncertainty. In *Uncertainty in Artificial Intelligence*.

A. Pfeffer, D. Koller, B. Milch, and K. T. Takusagawa. 1999. Spook: A system for probabilistic object-oriented knowledge representation. In *Uncertainty in Artifical Intelligence*.

C. Rich and C. Sidner. 1998. COLLAGEN: a Collaboration Manager for Software Interface Agents. *An International Journal: User Modeling and User Adapted Interaction*, 8.

M. Richardson and P. Domingos. 2006. Markov Logic Networks. *Machine Learning*, 62:107–136.

S. Srinivas and P. Nayak. 1996. Efficient Enumeration of Instantiations in Bayesian Networks. In *UAI*.

B. Thomson and S. Young. 2009. Bayesian Update of Dialogue State: A POMDP Framework for Spoken Dialogue Systems. *Computer Speech and Language*.

G. van Zanten. 1998. Adaptive Mixed-Initiative Dialogue Management. In *IEEE Workshop on Interactive Voice Technology for Telecommunications Applications*.

J. Williams. 2006. Partially Observable Markov Decision Processes for Dialog Management.

J. Williams. 2007. Using Particle Filters to Track Dialogue State. In *IEEE Workshop on Automatic Speech Recognition & Understanding*.

A Analysis of the Inference Algorithm

Theorem 1. *Algorithm 1 returns the top m MPEs of the POT along with their joint probabilities.*

Proof. We first prove this for the special case of $m = 1$ to simplify notation. For the base case of a node with no children, Algorithm 2 simply returns a message with all probabilities at 1 for all values of that node. Now, consider a node X with children $Y_1, \ldots, Y_k$. Let $\text{Desc}(Y)$ be the descendants of node Y. Since Algorithm 2 given node X returns exactly one explanation, $\boldsymbol{z}$ for each $x \in \mathcal{D}(X)$, we will define $\psi_X(x) = \psi_X(x, \boldsymbol{z})$. Now, to show that $\psi_X(x) = \max_{\text{Desc}(X)} \Pr(\text{Desc}(X)|X = x, E)$, that is, Algorithm 2 returns the top explanation of the entire subtree rooted at X for every value in $\mathcal{D}(X)$, we use structural induction on the tree.

$$
\begin{aligned}
&\max_{\text{Desc}(X)} \Pr(\text{Desc}(X)|X = x, E) \\
&= \max_{Y_{1:k},\text{Desc}(Y_{1:k})} \Pr(Y_{1:k}, \text{Desc}(Y_{1:k})|X = x, E) \\
&= \max_{Y_{1:k},\text{Desc}(Y_{1:k})} \prod_i \Pr(Y_i|X = x, E) \Pr(\text{Desc}(Y_i)|Y_i, E) \\
&= \prod_i \max_{Y_i,\text{Desc}(Y_i)} \Big[\Pr(Y_i|X = x, E) \Pr(\text{Desc}(Y_i)|Y_i, E)\Big] \\
&= \prod_i \max_{Y_i} \Big[\Pr(Y_i|X = x, E) \max_{\text{Desc}(Y_i)} \Pr(\text{Desc}(Y_i)|Y_i, E)\Big] \\
&= \prod_i \max_{Y_i} \Big[\Pr(Y_i|X = x, E)\psi_{Y_i}(y_i)\Big] \quad \{\text{Inductive step}\} \\
&= \psi_X(x).
\end{aligned}
$$

The proof for $m > 1$, where every maximization returns a list of the top m elements, is similar. □

B Dialogs in the Tourist Information Domain

Each user conducted 25 dialogs according to prescribed scenarios for the tourist information domain. The order of scenarios was randomized for each user. Sample scenarios:

1. Find a good and cheap Mexican restaurant in Mountain View.
2. There is a medical emergency and you need to get to the hospital. Find a route.
3. You need to find your favorite coffee franchise. You have 10 minutes to get coffee.
4. Find a place to buy some fruits and vegetables.
5. Find a Chinese restaurant in Santa Clara with good ambiance, and display travel distance.
6. Find an ATM on Castro Street in Mountain View.

Figure 7 shows a typical interaction with the system for the first scenario along with a possible hypothesis inferred by the system at every turn of the dialog. Figure 8 shows an example where the POT system is able to discard an incorrect observation about a restaurant based on the accumulated belief about bookstores over multiple turns. Figure 9 shows how the POT is able to leverage the ontological structure to pick out higher-level concepts with lower confidence scores over spurious low-level concepts with higher confidence scores.

User	Find a Mexican restaurant in Mountain View.
Hypothesis	*[venue restaurant] [area mountain view] [cuisine italian]* {Note: Mexican is misrecognized as Italian.}
User	No, Mexican.
Hypothesis	*[venue restaurant] [area mountain view] [cuisine mexican]*
User	Show me ones with at least four star rating.
Hypothesis	*[venue restaurant] [area mountain view] [cuisine mexican] [rating four star]*
User	I want a cheap place.
Hypothesis	*[venue restaurant] [area mountain view] [cuisine mexican] [rating four star] [price cheap]*
User	Is there anything on Castro?
Hypothesis	*[venue restaurant] [area mountain view] [street castro] [cuisine mexican] [rating four star] [price cheap]*

Figure 7: A sample dialog in the tourism information domain showing the inferred hypothesis of the user's intention at every turn. The information response from the system's back-end is based on its current hypothesis.

User utterance	Where is the bookstore?
ASR	`where is the bookstore`
True hypothesis	*[action venue] [venue store] [sell book]*
Baseline hypothesis	*[action venue] [venue store] [sell book]*
POT hypothesis	*[action venue] [venue store] [sell book]*
User utterance	Store on Market Street.
ASR	`store on market street`
True hypothesis	*[action venue] [venue store] [sell book] [street market]*
Baseline hypothesis	*[action venue] [venue store] [sell book] [street market]*
POT hypothesis	*[action venue] [venue store] [sell book] [street market]*
User utterance	In downtown.
ASR	`dennys`
True hypothesis	*[action venue] [venue store] [sell book] [street market] [area downtown]*
Baseline hypothesis	*[action venue] [venue restaurant] [brand dennys]*
POT hypothesis	*[action venue] [venue store] [sell book] [street market]*

Figure 8: A dialog showing the ASR input for the user's utterance, and the corresponding true, baseline, and POT hypotheses. The POT is able to correctly discard the inconsistent observation in the third turn with the observations in previous turns.

User utterance	Where should I go to buy Lego for my kid?
SU slots	⟨Venue Store 38⟩ ⟨ServiceType GolfCourse 60⟩
True hypothesis	*[action venue] [venue store] [storetype toy]*
Baseline hypothesis	*[action venue] [venue service] [servicetype golf course]*
POT hypothesis	*[action venue] [venue store]*

Figure 9: A single dialog turn showing the SU slots for the user's utterance, and the corresponding baseline, POT, and true hypotheses. Any system that looks at the individual confidence scores will base its hypothesis on the ⟨ServiceType GolfCourse 60⟩ slot. Instead, the POT hypothesis is influenced by ⟨Venue Store 38⟩ because its score in combination with the concept's location in the POT makes it more likely than the other slot.