Layered Semantic Graphs for Dialogue Management

Jiaying Shen
Nuance Communications
Sunnyvale, CA
jiaying.shen@nuance.com

Hendrik Harkema
Nuance Communications
Sunnyvale, CA
hendrik.harkema@nuance.com

Richard Crouch
Amazon
Palo Alto, CA
rcrouch@amazon.com

Peng Yu
Mobi Systems
Cambridge, MA
peng@takemobi.com

Ciaran O'Reilly
Nuance Communications
Portland, OR
ciaran.oreilly@nuance.com

Abstract

This paper proposes a layered semantic graph representation for dialogue information. The representation factors information into several interdependent layers, facilitating efficient information access and processing by the components in a dialogue system. We describe the layers in the semantic graph and the function they serve in an implemented task-oriented dialogue system.

1 Introduction

At Nuance Communications we are developing a conversational system for multi-turn task-oriented dialogues. The system plays the role of a virtual concierge, assisting the user with such tasks as finding restaurants, parking, and making reservations. Interactions between system and user are flexible, supporting cross-domain, multi-intent search dialogues and allowing for the addition or revision of constraints at any point in the exchange. Linguistically, users can express themselves in a natural way to the system, using anaphoric expressions, asking Wh-questions, and using logical operators such as conjunction, disjunction, and negation to build complex search constraints. The system is also capable of reasoning with temporal and spatial constraints between events.

A challenge in building dialogue systems with this level of complexity is managing the diverse kinds of information flowing through them, such as the interpretation of natural language input, current task focus, query results from knowledge sources, and the temporal order of events. We propose using layered semantic graphs for this purpose. As a unifying graph representation, its layers are subgraphs representing specific aspects of information relevant to dialogue management. The layers are connected and are incrementally augmented as the dialogue unfolds. The result is a single, uniform, graphical representation of dialogue information that can be traversed and manipulated by a dialogue manager using known graph methods. It can be easily extended to new types of information by adding new layers. Also, the graph formalism naturally aligns with existing syntactic and semantic representations such as dependency structures and knowledge graphs.

Layered semantic graphs facilitate complex information processing steps in dialogue understanding. They enable canonicalization, which abstracts away from syntactic variation in user requests that doesn't affect meaning. They help bridge structural differences between the linguistic input and backend knowledge resources, which is necessary for interpreting user input in terms of the capabilities of the system. The graphs also support the integration of diverse inputs and outputs for reasoning components, as well as simple backtracking to address conflicts or inconsistencies that may arise during a dialogue.

In the rest of this paper, following a discussion of related work in section 2, we provide a detailed description of the semantic graph layers used in our dialogue system. Section 4 discusses the versatility and expressivity of the approach. The paper closes with conclusions and directions for future work.

2 Related Work

The idea of layered semantic graphs for dialogue management naturally arose out of a proposal for a logic of concepts and contexts (de Paiva et al., 2007; Bobrow et al., 2005). This logical system provides a semantics for natural language that distinguishes between conceptual and contextual structure. The concepts and relationships between concepts making up the conceptual structure indicate the predicate-argument structure of a sentence, i.e., "who does what to whom". The contextual structure layered on top of the conceptual structure is concerned with instantiability of concepts, e.g., concepts occurring in negation contexts are asserted not to have instances. Separating and layering the different structures in the semantic representation facilitates reasoning with the meaning of sentences and world knowledge for tasks like textual inference (Bobrow et al., 2007; Boston et al., forthcoming).

The layered semantic graph approach is also related to the correspondence architecture of lexical functional grammar (Kaplan, 1995; Asudeh, 2012). This architecture defines several levels of linguistic representation, related to one another by correspondence functions that map between elements on different levels. The separation into levels allows for the formulation of "modular" linguistic generalizations which govern a given level independently from others. Analogously, our semantic graphs factor dialogue information into several interdependent layers for use by the various components in our dialogue system.

This explicit organization of information contrasts with the "latent" representations used in end-to-end deep learning approaches to dialogue, e.g., Eric et al. (2017), Bordes and Weston (2017). One could, however, imagine a neural dialogue parser that predicts the different types of information in the graph, similar to Bapna et al. (2017). Factorization of information affords data efficiency in the sense that each dialogue task (intent recognition, query formulation, etc.) can be learned independently.

Graph-based structures are ubiquitous in dialogue research. They are used to characterize the architecture and information flow within dialogue systems, e.g., Schlangen and Skantze (2009), to represent dialogue state, e.g., Ramachandran and Ratnaparkhi (2015), and to structure background knowledge, e.g., Hixon et al. (2015). Similarly, various probabilistic graphical modeling languages have been used to provide compact and expressive representations of domain knowledge for tracking dialogue state, e.g., Lison (2015), or integrating multiple information sources to infer intent, e.g., Kenington and Schlangen (2014). Our work differs from these approaches in that it doesn't focus on the operation of specific dialogue components or the overall architecture. Instead, this paper addresses the practical yet rarely discussed concern of representing and integrating diverse information within a dialogue system.

More closely related to our paper, the TRIPS dialogue system (Allen et al., 2005) proposes an intermediary representation (AKRL) to connect natural language processing output to backend representations. The layered semantic graph differs from AKRL in several meaningful ways: it does not restrict the implementation of individual components, it encodes information produced by components other than just natural language understanding and the backend, and it is cumulative across turns in a dialogue.

3 Layered Semantic Graphs

In a layered semantic graph, the linguistic meaning representation layers are based on the conceptual and contextual structures discussed in the previous section. To these, several new layers essential for managing dialogues were added. Following Kalouli and Crouch (2018), the linguistic layers include a role layer, for predicate-argument structure; a context layer, for logical operators and other clausal contexts; a lexical layer, for conceptual and ontological information; and a link layer, for coreference and discourse links. The dialogue-specific layers include a query layer, for queries to backend knowledge bases; a knowledge layer, for the results returned by these queries; and several planning-related layers, for temporal relations between multiple events.[1] Each layer is composed of edges unique to that layer and the nodes they connect. The same node may appear in multiple layers, but not so the edges.

All these layers together enable our system to reason with the meaning of dialogue utterances and perform dialogue interpretation tasks such as intent and mention recognition, temporal reasoning, and

[1]The semantic graphs in our implemented system have several additional layers which are not discussed in this paper.

backend query formulation. In the rest of this section, we will describe the linguistic and dialogue layers in more detail, as well as the role they play in dialogue interpretation.

3.1 Linguistic Layers

The linguistic layers represent various aspects of the meaning of user utterances in a dialogue. Our system uses "deep" natural language understanding, provided by the Cognition system (Goldsmith et al., 2009; Dahlgren, 2013), relying on meaning representations that provide more finesse than flat intent and mention structures, in order to capture complex logical relations between mentions and intents and to support the representation of questions. An input utterance is first parsed, resulting in a syntactic structure that provides the basis for determining the scope of negation, quantifiers, and referential expressions. Next, a logical form, akin to a first-order logical formula and adhering to a neo-Davidsonian view of events (Davidson, 1980; Moltmann, 2015), is derived from this structure, and then translated into the linguistic layers.

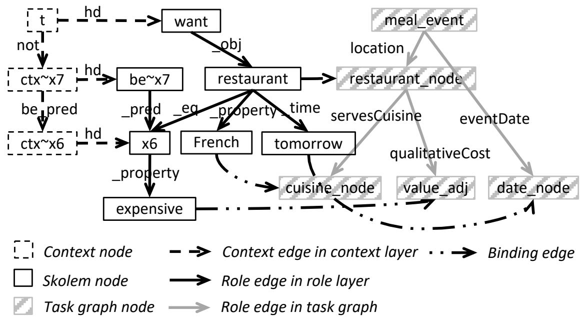

Figure 1: Linguistic layers for "I want a French restaurant for tomorrow that is not expensive"

Role layer: The role graph expresses the basic propositional content of an utterance. Its member skolem nodes correspond to the unary predicates in the logical form, which generally arise from content words in the input utterance, and assert the existence of concepts. This layer makes no claims as to the existence of instances of these concepts. The edges are provided by the binary and higher arity predicates in the logical form, encoding the semantic relationships between words in the sentence.

For example, the role graph for the user utterance "I want a French restaurant for tomorrow that is not expensive" is given in figure 1.[2] The "_eq" edge between the skolem nodes labeled "restaurant" and "x6" equates the two nodes: propositionally, the restaurant is French, expensive, and for tomorrow. The negation of "expensive" is handled in the context layer.

Lexical layer: The lexical layer associates skolem nodes with entries in the Cognition semantic lexicon (Dahlgren, 1988). Most importantly, the lexical information for skolem nodes includes disambiguated word senses that are attached to concepts in the Cognition ontology (ibid.). Technically, the lexical layer consists of edges labeled "lex" connecting the skolem nodes in the role graph and a set of sense nodes, holding the lexical information.[3] For example, the skolem node "French" in the example sentence is associated with the word sense "French-1", defined as "of France" in the Cognition lexicon. Another possible word sense, not selected here, is "French-2", referring to the French language.

The information in the lexical layer is crucially important for interpreting a user utterance in terms of the tasks that the system can perform. Each task is represented as a graph whose nodes are also taken from the Cognition ontology. Such a task graph constitues a "mini-ontology of mentions", specifying how a user may talk about a task. For example, the simplified task graph on the right in figure 1, for making restaurant reservations, shows a node labeled "restaurant_node" that is linked to a node labeled "cuisine_node" through an edge labeled "servesCuisine", as restaurants typically serve a specific cuisine, and users are likely to mention restaurants and cuisines when making restaurant reservations. Now, the cuisine node in the task graph binds the "French" node in the role graph because in the ontology the concept "nationality_group", which is lexically associated with the skolem node "French" through its word sense "French-1", is a subconcept of the concept "cuisine_node". A binding like this counts as positive evidence for a restaurant reservation interpretation of the example sentence. Note that the negation of "expensive", which is not part of the role graph, is irrelevant for the purposes of binding; a sentence like "I want a restaurant that is not expensive" is as much about restaurant reservations as a sentence like "I want a restaurant that is expensive".

[2]For practical reasons, the translation step from logical form to semantic graph ignores certain lexical heads, such as the personal pronouns "I" and "you"; therefore the node labeled "want" in the role graph is lacking a subject.

[3]The sense nodes are not displayed in figure 1.

Context layer: Currently, the main function of the context layer is to record the scope of (nested) logical operators in user utterances, specifically conjunction, disjunction, and negation. The interpretation of Wh-questions also relies on contexts. Contexts are represented by context nodes in the context graph. Every context graph has a top or "true" context. Additional contexts are nested below the top context. In the graph, nesting of contexts is represented by edges between context nodes. The label of the incoming edge (for context nodes other than top) indicates the nature of the context (e.g., "not" for a negation context). Every context has a head; this relationship is marked by edges labeled "hd" from a context node to a skolem node in the role graph. The head defines the extent or scope of the context. For example, there are three contexts in figure 1, nested in this order: the top context (node "t"), present by default; a context for negation (node "ctx∼x7"), introduced by "not"; and a context for the predication associated with "is" (node "ctx∼x6"). Informally, as indicated by their heads, the scope of the "true" context is "want a French restaurant for tomorrow", and the negation includes the node for "is" and the predication that "x6" (equated with the restaurant) has the property "expensive".

The context layer is also used for backend query formulation. For example, as shown in figure 1, the node labeled "expensive" is bound to the task graph node labeled "value_adj", which eventually hooks into the "cost" field of a restaurant query. Because "expensive" appears in a negative context in the context graph, the relevant query term is to be negated in the query.

Link layer: The link graph is the locus of information about identities between nodes in the role graph as induced by anaphora resolution. Inter- and intra-sentential anaphora, potentially across dialogue turns, are resolved by the Cognition parser following the approach of Lee et al. (2013). These coreferences are modeled in the link graph as edges between skolem nodes. The dialogue manager is able to identify additional coreferences between mentions in user utterances and the results returned by backend queries, as in, for example, a situation in which the dialogue manager proposes a restaurant to the user and they subsequently ask, "When is *it* open"? Coreferences of this kind exist in the link graph as edges between skolem nodes and knowledge nodes in the knowledge layer (see section 3.2). The contents of the link graph factor into the interpretation of user utterances vis-à-vis the library of task graphs. An edge in the link graph is interpreted as a signal to restrict the bindings of the anaphoric expression (a skolem node) to the bindings dictated by its antecedent (a skolem node or knowledge node). Bridging anaphora, in which an anaphoric expression indirectly refers to another expression, e.g., Nand and Yeap (2013), are also encoded in the link layer.

3.2 Dialogue Layers

In addition to the linguistic layers, the semantic graph has been extended to include several novel dialogue layers that assemble and keep track of information gathered from knowledge sources as well as dialogue decisions made by various reasoning components in the system.

Query layer: The linguistic layers of a semantic graph represent linguistic meanings. However, for a couple of reasons, they cannot be used directly to form backend queries. First, the word senses in the lexical layer and the relations between the skolem nodes in the role graph are often not specific enough. For example, in the utterance "I want a French restaurant for tomorrow that is not expensive", "French" corresponds to the word sense "French-1" in the lexical layer, meaning "of France". Similarly, in the role graph, the relation between "French" and "restaurant" is a generic "_property" relation (see figure 1). Without further reasoning, we have no way of knowing that "French" refers to the cuisine served by the restaurant, rather than to its location or the nationality of the owner. Secondly, the syntactic structure of a sentence, and hence the role graph derived from it, does not always accurately reflect the underlying ontological relations between query entities and their attributes. For example, in the role graph for the sample sentence, "tomorrow" modifies "restaurant". However, for the purposes of query formulation, "tomorrow" is an attribute of a meal event that is not explicitly expressed in the utterance.

To address these issues, a query layer is added to encode world knowledge concepts and relationships. The nodes and edges in the query layer mirror the structure of the task graphs discussed earlier. Dialogue interpretation uses this correspondence, plus the bindings between the task graphs and the role layer, to bind query nodes in the query layer to skolem nodes in the role layer. These bindings reconcile the linguistic information with world knowledge. The query layer also helps to abstract away from lexical and syntactic variation in utterances, i.e., variation in the linguistic layers that does not change the interpretation.

For example, figure 2 shows the bindings between the role layer and the query layer for the example utterance. Here, “French” is bound to “cuisine_node” and “restaurant” to “restaurant_node”. The edge between “cuisine_node” and “restaurant_node”, i.e., “servesCuisine”, provides a more specific relation for “French” and “restaurant” than the linguistic “_property”. Also, “tomorrow” is bound to “date_node”, which, as desired, modifies “meal_event” via the “eventDateIs” relation, supplanting the linguistic attachment of “tomorrow” to “restaurant” in the role graph. Notice also that the query layer has its own context nodes, derived from the linguistic context layer.[4] This is necessary since the relations in the query layer are not in a one-to-one correspondence to those in the linguistic layers.

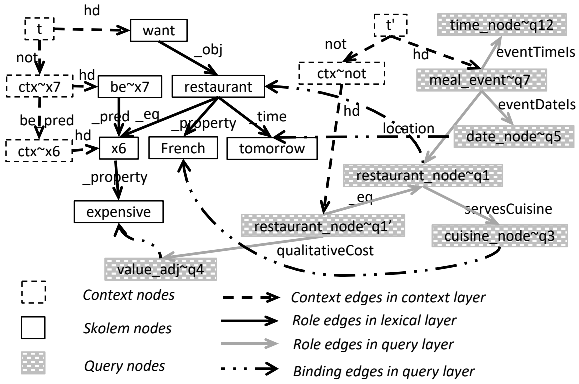

Figure 2: Query layer for “I want a French restaurant for tomorrow that is not expensive”

The query layer is used to construct well-formed queries that can be understood by backend knowledge bases. Towards that end, a query reasoner is called to fill out the query layer with additional nodes and relations. For example, though the user did not specify a desired time, the reasoner added a new query node “time_node∼q12” to the query layer, because the system needs to have a restaurant reservation time in order to return a useful answer to the user. Figure 2 shows the complete query graph for the sample utterance after the query reasoner has been called.

Knowledge layer: The query results returned from the knowledge base are integrated into the semantic graph via the knowledge layer. Figure 3 shows the knowledge layer for the running example. The query node “meal_event∼q7” is grounded in the top level knowledge node “Left Bank Santana Row at 19:00 on 2018-3-5”. The attribute query nodes, e.g., “time_node”, are grounded in their values, e.g., “19:00”. Additionally, relations between grounded instances are recorded as well, e.g., the role edge “eventTimeIs” between the knowledge nodes representing “Left Bank Santana Row at 19:00 on 2018-3-5” and “19:00”. The knowledge layer allows the dialogue manager to keep track of the current options available to the conversation and to change them dynamically as the dialogue unfolds. It also links the grounded entities to their attribute values.

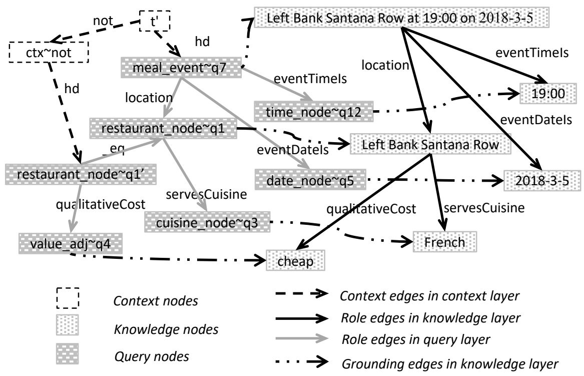

Figure 3: Knowledge layer for “I want a French restaurant for tomorrow that is not expensive”

Planning layer: Our dialogue system can handle requests from the user to schedule events that are temporally or spatially dependent, e.g., “Find an Italian restaurant for two people tonight. I also want to see a comedy movie after that”. While the knowledge base can supply candidates for Italian restaurants available at the requested time as well as movie show times, an AI planner is needed to deal with the temporal and spatial relations between the two events in order to arrive at a cohesive plan.

The planner can retrieve all event-related information, including the event candidates and their locations and times, directly from the query and knowledge layers. However, the temporal relations expressed in the linguistic layers are often not precise enough. We add a planning layer to address this problem.

[4]An explanation of this derivation is beyond the scope of this paper.

The edges in the planning layer connect the query nodes representing events to be scheduled and are generated by a commonsense reasoner that maps linguistic temporal relations onto Allen relations (Allen, 1983; Dechter et al., 1991). For example, in figure 4, the "after" relation between "movie" and "restaurant" is translated into the Allen relation "precedes" from "meal_event" to "movie_event". By combining information from the query, knowledge, and planning layers, a planning problem can be generated.

Solution layer: When the planner finds a satisfactory plan, it is encoded in the solution layer. The assignment edges connect the query nodes representing the events to be scheduled to one of their grounded knowledge nodes, meaning that this particular assignment is part of the solution. For example, the solution generated for the request "comedy movie after Italian restaurant" includes the assignments of "Lady Bird at AMC Mercado 20 at 9:25 pm on 2018-3-5" to the movie event, and "Rulfo at 7:00 pm on 2018-3-5" to the meal event, as shown in figure 4.

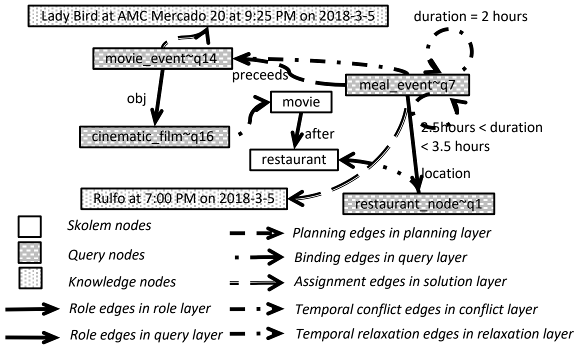

Figure 4: Planning layers for "comedy movie after Italian restaurant"

Conflict and relaxation layers: The planner cannot always find a perfect plan satisfying all user requirements. When confronted with an over-subscribed problem, the planner tries to suggest an alternative solution by relaxing some temporal or domain constraints (Yu et al., 2016a; Yu et al., 2016b). An example of a temporally relaxed recommendation is "You wanted a movie after your restaurant reservation tonight. Since typically your restaurant reservation lasts between 2.5 hours and 3.5 hours, I cannot find a plan. However, if you shorten the time to 2 hours, Rulfo is available at 7:00 pm today. Then Lady Bird is showing at AMC Mercado 20 at 9:25 pm. Is that ok?". Here the system presents the temporal conflicts that render the original planning problem as stated by the user unsolvable. Then it suggests shortening the meal event and presents the resulting plan.

In order to keep track of this information, we add conflict and relaxation layers to the semantic graph. The conflict layer encodes the temporal conflicts, either as a duplicate of the planning edge representing the temporal constraint causing the conflict, e.g., the "precedes" relation from "meal_event" to "movie_event" in figure 4, or as a new temporal conflict edge representing a default constraint, e.g., the duration of the "meal_event" in the same figure. A temporal relaxation is represented as an edge that is similar to the planning edge representing the original temporal constraint, but relaxed. Figure 4 shows the temporal relaxation of the meal event duration to 2 hours.

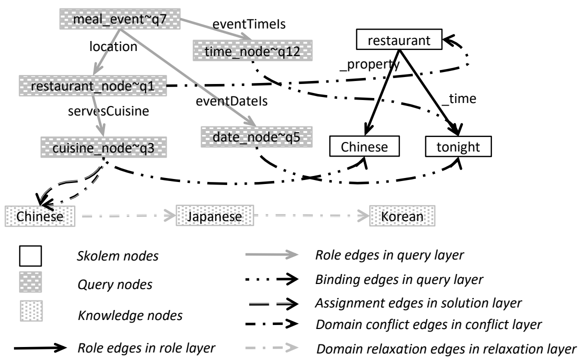

Figure 5: Domain relaxation for "find a Chinese restaurant"

When temporal relaxation is not sufficient to find a solution, it may be preferable to relax a domain constraint instead. For example, when the system can't find a Chinese restaurant at the requested time, it may suggest a Japanese restaurant instead. Here a domain conflict is represented in the conflict layer as a domain conflict edge, which is essentially a duplicate of the original grounding edge representing the domain constraint causing the conflict. As an example, in figure 5 there is a domain conflict edge between "cuisine_node" and the knowledge node "Chinese", meaning the constraint of "Chinese restaurant" is what rendered the problem unsolvable. Since a domain relaxation replaces the value of a domain constraint, a domain relaxation edge is an edge from the knowledge node representing the attribute value being relaxed (e.g., "Chinese") to a knowledge node representing the newly suggested value (e.g., "Japanese"). The user may reject the suggested domain relaxation, causing the system to suggest yet another value. The relaxation chain is then extended by adding another domain relaxation edge from the last relaxed knowledge node to a new one (e.g., "Korean"), as shown in figure 5. This flexible representation allows the planner to freely explore the relaxation search space and enables the dialogue manager to keep track of the relaxation paths and retract a previously relaxed constraint if needed.

4 Assessment

Since this paper focuses on representation rather than processing, we chose not to include an extrinsic evaluation on some dialogue task. Instead, in this section we assess the versatility and expressivity of layered semantic graphs. Versatility means the graphs impose no constraints on the formalisms used in the components of the dialogue system. This will be demonstrated for the linguistic layers and the query layer. Regarding expressivity, we will compare the planning layer with other planning languages in terms of their ability to capture information pertinent to solving planning problems. We will also give an example of a multi-turn, multi-intent dialogue to illustrate how layered semantic graphs are applicable beyond single shot scenarios and accumulate information throughout a conversation.

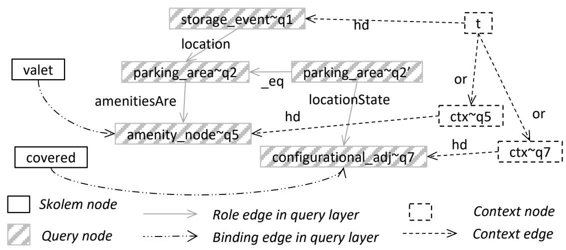

Figure 6: Layered semantic graph for statistical NLU output of "Find valet or covered parking"

One of the goals of layered semantic graphs is the ability to encode information produced by different components in a dialogue system, regardless of their underlying implementation. One example of this versatility can be found in the linguistic and query layers. So far, we have focused on a first-order logic representation as the output of the NLU component in our dialogue system. However, NLU approaches based on statistical methods and machine learning are also widely used in spoken dialogue systems, and commonly employ semantic frame based representations (Wang et al., 2011). For example, the semantic frame for "Find valet or covered parking" may look like this: {*"nluSlots": {"INTENTION": ["search_parking"], "type": [{"OR": ["covered","valet"]}], "relative_location": "near"*}}. Here the attribute-value pairs in the semantic frame essentially are the bindings between the skolem nodes in the user utterance and the query nodes in the query graph. Similarly, the nested logical operator "OR" directly corresponds to the context nodes in the context layer. We have implemented a translation method for an existing statistical NLU component, which for the example sentence outputs the layered semantic graph given in figure 6. The linguistic layers in this graph are much more simplistic than those resulting from deep NLU, as the layered semantic graph is merely a representation of the outputs from the components in the dialogue system. In a similar fashion, one can define more complex translations into a graph's linguistic layers from semantic representation languages such as AMR (Banarescu et al., 2013) and more application-specific formalisms like AMRL (Kollar et al., 2018).[5]

Another objective of the semantic graph is to encode planning problems while preserving the semantic meanings behind all the task and constraint models. Many languages exist for encoding planning problems. PDDL ((McDermott et al., 1998)) is an early and widely used formalism, and its latest developments support a large set of features, such as temporal constraints (Fox and Long, 2003), non-linear objectives (Gerevini and Long, 2005), and probabilistic effects (Younes and Littman, 2004). However, designed as abstract formalisms for describing planning domains, they are unable to preserve the semantic meanings or the mapping with dialogue inputs, yet these are key features for the dialogue manager to function. Prior work on planner-based dialogue management systems require extra translation and state-keeping layers to fill the gap (Allen et al., 2001). Layered semantic graphs encode temporal planning problems using time-evolved goals, and use a single model for both the planning domains and the problems. The approach also supports a rich set of temporal constraints from the STN (Dechter et al., 1991) and STNU (Morris et al., 2001) formalisms to more precisely model temporal relations.

[5]For a comparison of the semantic formalism underlying our original linguistic layers to other semantic parsing representations, the reader is referred to Kalouli and Crouch (2018).

The semantic graph model is not mutually exclusive with existing planning languages: many features in PDDL, RDDL and RMPL can be incorporated into it. For example, in order to generate robust travel plans in real-world traffic, we have extended the temporal constraint encoding to model temporal uncertainty using set-bounded (from the STNU formalism) and probabilistic approaches (from the pSTN formalism, Santos Jr and Young (1999)). Improving the expressivity of semantic graphs for preference, uncertainty and multi-agent modeling is key for many applications in the dialogue management field, and is part of our future work.

One important requirement for the semantic graph is to be able to accumulate information across a multi-turn, multi-intent dialogue. Consider the following multi-turn variation of the user request in figure 4: *User*: "Find an American restaurant for two people tonight." *System*: "Lion and Compass is available at 7:20 pm today. Is that ok?" *User*: "Actually I want an Italian restaurant." *System*: "Rulfo is available at 7:00 pm today. Is that ok?" *User*: "I also want a comedy movie after that." Figure 7 shows the resulting semantic graph for this dialogue. The date_node and time_node are still bound to "tonight" in the first user utterance. However, the cuisine_node is no longer bound to "American" but instead to "Italian" specified in the second user utterance. Additionally, "that" in the third user utterance is anaphorically linked to the knowledge node proposed by the system in the previous system utterance. This is a perfect example of how information can be resolved and accumulated in a consistent manner and preserved in the semantic graph.

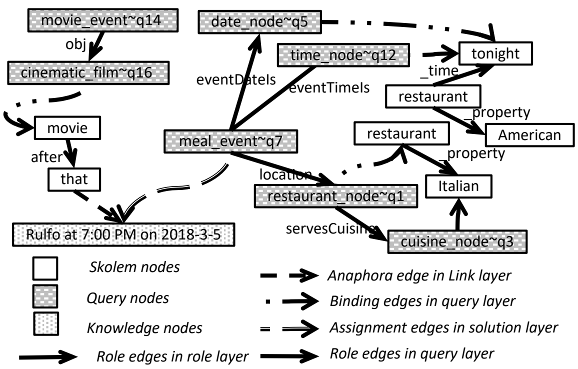

Figure 7: Layered semantic graph for the multi-turn multi-intent dialog

5 Conclusion

In this paper, we have presented a layered semantic graph that provides a unified graphical representation of various types of information flowing through a complex task-oriented dialogue system. The graph is expressive, containing a wide range of linguistic information extracted from user utterances, as well as keeping track of non-linguistic information produced by the knowledge sources and reasoners used in the system. All this information, though diverse, is intimately interrelated. We have also illustrated the versatility of the approach: the use of layered semantic graphs is not tied to specific implementations or internal representations of the dialogue components.

Instead of accumulating all dialogue information into a single monolithic representation, we explicitly factor it into layers according to the unique characteristics of each reasoner in the dialogue system. This allows for a modular separation of information, while preserving the connections between the layers, making finding, accessing, and processing information more tractable. Each reasoner only needs to look in the relevant layers to find the data it needs. Its output, in turn, can easily be integrated into the graph, with a clear delineation of consistency between the layers. Additionally, when information or recommendations need to be retracted, the chain of reasoning can be traced back across the layers.

The layered semantic graph is also extensible. In this paper, we have described the layers we need to support the functionality in scope for our task-oriented dialogue system. Other dialogue settings, e.g., multi-agent tasking, require additional richness. When building a dialogue system, we can add new layers to the graph to accommodate new reasoning components, keeping the information flow smooth and consistent across the system. This flexibility is a powerful feature for practical dialogue system engineering. It has become a central part of the dialogue state in our system, and has proven essential in being able to carry on a consistent, flexible, natural and complex dialogue with the user.

For future work, we plan to build a better visualization toolkit for the graph in order to aid in system building, debugging, and information display. We also plan to explore the possibility of encoding and reasoning with other contextual information in the context layer, such as propositional attitudes.

References

James Allen, George Ferguson, and Amanda Stent. 2001. An architecture for more realistic conversational systems. In *Proceedings of the 6th international conference on Intelligent user interfaces*, pages 1–8. ACM.

James F. Allen, George Ferguson, Amanda Stent, Scott C. Stoness, Mary D. Swift, Lucian Galescu, Nathanael Chambers, Ellen Campana, and Gregory Aist. 2005. Two diverse systems built using generic components for spoken dialogue (recent progress on TRIPS). In *ACL*, pages 85–88. The Association for Computer Linguistics.

James F. Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):pp.832–843, November.

Ash Asudeh. 2012. *The Logic of Pronominal Resumption*. Oxford University Press, Oxford, UK.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria, August. Association for Computational Linguistics.

Ankur Bapna, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2017. Sequential dialogue context modeling for spoken language understanding. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 103–114. Association for Computational Linguistics.

Daniel G. Bobrow, Cleo Condoravdi, Richard Crouch, Ronald M. Kaplan, Laurie Karttunen, Tracy Holloway King, Valeria de Paiva, and Annie Zaenen. 2005. A basic logic for textual inference. In *Proceedings of the AAAI Workshop on Inference for Textual Question Answering*.

Daniel G. Bobrow, Cleo Condoravdi, Richard Crouch, Valeria de Paiva, Laurie Karttunen, Tracy Holloway King, Rowan Nairn, Charlotte Price, and A. Zaenen. 2007. Precision-focused textual inference. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 16–21.

Antoine Bordes and Jason Weston. 2017. Learning end-to-end goal-oriented dialog. In *Proceedings of the 5th International Conference on Learning Representations*.

Marisa Ferrara Boston, Richard Crouch, Erdem Özcan, and Peter Stubley, forthcoming. *A Festschift for Lauri Kartunnen*, chapter Natural Language Inference Through a Semantic Lexicon and Ontology. CSLI Publications, Stanford University.

Kathleen Dahlgren. 1988. *Naive Semantics for Natural Language Understanding*. Kluwer Academic Press, Norwell, MA.

Kathleen Dahlgren. 2013. Formal linguistic semantics and dialogue. In *Proceedings of the Annual Semantic Technology Conference*, SemTech.

Donald Davidson. 1980. *Essays on Action and Events*. Clarendon, Oxford, UK.

Valeria de Paiva, Daniel G. Bobrow, Cleo Condoravdi, Richard Crouch, Lauri Karttunen, Tracy Holloway King, Rowan Nairn, and Annie Zaenen. 2007. Textual inference logic: Take two. In *Proceedings of the 3rd International Workshop on Contexts and Ontologies: Representation and Reasoning*.

Rina Dechter, Itay Meiri, and Judea Pearl. 1991. Temporal constraint networks. *Artificial Intelligence*, 49(1-3):61–95, May.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 37–49.

Maria Fox and Derek Long. 2003. Pddl2. 1: An extension to pddl for expressing temporal planning domains. *Journal of artificial intelligence research*, 20:61–124.

Alfonso Gerevini and Derek Long. 2005. Plan constraints and preferences in pddl3. Technical report, 2005-08-07, Department of Electronics for Automation, University of Brescia, Brescia, Italy.

Elizabeth J. Goldsmith, Saurabh Mendiratta, Radha Akella, and Kathleen Dahlgren. 2009. Natural language query in the biochemistry and molecular biology domains based on Cognition search. In *Proceedings of the 2009 Summit on Translational Bioinformatics*, pages 32–37.

Ben Hixon, Peter Clark, and Hannaneh Hajishirzi. 2015. Learning knowledge graphs for question answering through conversational dialog. In *HLT-NAACL*, pages 851–861. The Association for Computational Linguistics.

Aikaterini-Lida Kalouli and Richard Crouch. 2018. GKR: the graphical knowledge representation for semantic parsing. In *Proceedings of the Workshop on Computational Semantics beyond Events and Roles*, pages 27–37, New Orleans, Louisiana, June. Association for Computational Linguistics.

Ronald M. Kaplan, 1995. *Formal Issues in Lexical-Functional Grammar*, chapter The Formal Architecture of Lexical-Functional Grammar. CSLI Publications, Stanford University.

Casey Kennington and David Schlangen. 2014. Situated incremental natural language understanding using markov logic networks. *Computer Speech & Language*, 28(1):240 – 255.

Thomas Kollar, Danielle Berry, Lauren Stuart, Karolina Owczarzak, Tagyoung Chung, Lambert Mathias, Michael Kayser, Bradford Snow, and Spyros Matsoukas. 2018. The Alexa meaning representation language. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 177–184, New Orleans - Louisiana, June. Association for Computational Linguistics.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Journal of the Association for Computational Linguistics*, 39(4):885–916.

Pierre Lison. 2015. A hybrid approach to dialogue management based on probabilistic rules. *Computer Speech & Language*, 34(1):232 – 255.

Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. 1998. Pddl—the planning domain definition language. Technical report, CVC TR98003/DCS TR1165. New Haven, CT: Yale Center for Computational Vision and Control.

Friederike Moltmann, 2015. *Oxford Handbook of Event Structure*, chapter Nominals and Event Structure. Oxford University Press, Oxford, UK.

Paul Morris, Nicola Muscettola, and Thierry Vidal. 2001. Dynamic control of plans with temporal uncertainty. In *Proceedings of the 17th international joint conference on Artificial intelligence (ICAI'01)*, pages 494–499, August.

Parma Nand and Wai Yeap, 2013. *Agents and Artificial Intelligence; Communications in Computer and Information Science, vol. 358*, chapter A Framework for Interpreting Bridging Anaphora. Springer-Verlag, Berlin, Heidelberg.

Deepak Ramachandran and Adwait Ratnaparkhi. 2015. Belief tracking with stacked relational trees. In *SIGDIAL Conference*, pages 68–76. The Association for Computer Linguistics.

Eugene Santos Jr and Joel D Young. 1999. Probabilistic temporal networks: A unified framework for reasoning with time and uncertainty. *International Journal of Approximate Reasoning*, 20(3):263–291.

David Schlangen and Gabriel Skantze. 2009. A general, abstract model of incremental dialogue processing. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 710–718, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ye-Yi Wang, Li Deng, and Alex Acero. 2011. *Semantic Frame Based Spoken Language Understanding*. Wiley, January.

Håkan LS Younes and Michael L Littman. 2004. Ppddl1. 0: The language for the probabilistic part of ipc-4. In *Proc. International Planning Competition*.

Peng Yu, Jiaying Shen, Peter Z. Yeh, and Brian Williams. 2016a. Resolving over-constrained conditional temporal problems using semantically similar alternatives. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 3300–3307. AAAI Press.

Peng Yu, Jiaying Shen, Peter Z. Yeh, and Brian Williams. 2016b. Towards personal assistants that can help users plan. In David Traum, William Swartout, Peter Khooshabeh, Stefan Kopp, Stefan Scherer, and Anton Leuski, editors, *Intelligent Virtual Agents*, pages 424–428. Springer International Publishing.