Tackling Winograd Schemas by Formalizing Relevance Theory in Knowledge Graphs

Peter Schüller
Department of Computer Engineering, Marmara University, Goztepe Kampusu
34722 Kadikoy, Istanbul, Turkey
schueller.p@gmail.com

Abstract

We study disambiguating of pronoun references in Winograd Schemas, which are part of the Winograd Schema Challenge, a proposed replacement for the Turing test. In particular we consider sentences where the pronoun can be resolved to both antecedents without semantic violations in world knowledge, that means for both readings of the sentence there is a possible consistent world. Nevertheless humans will strongly prefer one answer, which can be explained by pragmatic effects described in Relevance Theory. We state formal optimization criteria based on principles of Relevance Theory in a simplification of Roger Schank's graph framework for natural language understanding. We perform experiments using Answer Set Programming and report the usefulness of our criteria for disambiguation and their sensitivity to parameter variations.

Introduction

The Winograd Schema Challenge (Levesque, Davis, and Morgenstern 2012) is an anaphora resolution task which was proposed as a challenging but more reasonable alternative for the Turing Test. The challenge has received interest in the Artificial Intelligence community and beyond (Marcus 2013) as a chance to push forward natural language *understanding* capabilities of computers.

Winograd Schemas (WSs) are pairs of sentences that are referential ambiguous, i.e., they can be semantically interpreted in at least two distinct ways. Only one of these interpretations is correct and humans easily identify the correct one.

$$\begin{array}{l}\text{There is [a gap]}_g \text{ in [the wall]}_w.\\ \text{You can see the garden through it}_{w,g?}.\end{array} \quad (1)$$

$$\begin{array}{l}\text{There is [a gap]}_g \text{ in [the wall]}_w.\\ \text{You can see the garden behind it}_{w,g?}.\end{array} \quad (2)$$

(1) and (2) show such a pair of sentences, where we denote phrases with square brackets and coreference using subscripts. Here the pronoun 'it' is ambiguous between noun phrases 'a gap' and 'the wall'. In (1) the solution is 'it_w' — we can see the garden through the gap, in (2) the solution is 'it_g' — we can see the garden behind the wall. Note that only a single word differs between (1) and (2). This WS can be disambiguated by reasoning over world knowledge such as 'walls are solid objects', 'solid objects are nontransparent', and 'one cannot see through nontransparent objects'. An interesting challenge of WSs is that they seem to *require deep reasoning* beyond the surface of the text, in particular beyond the current state of the art of (mostly statistical) natural language processing technology.

Some schemas can be disambiguated using linguistic theories of text structure, for example the following.

$$\begin{array}{l}\text{[Pete]}_p \text{ envies [Martin]}_m\\ \text{because he}_{p,m?} \text{ is very successful.}\end{array} \quad (3)$$

$$\begin{array}{l}\text{[Pete]}_p \text{ envies [Martin]}_m\\ \text{although he}_{p,m?} \text{ is very successful.}\end{array} \quad (4)$$

In (3) the word 'because' indicates that '$\text{he}_{p,m?}$ is very successful' is a plausible reason for 'Pete envies Martin'. As a result we can rule out the coreference 'he_p' because Martin's success is a more plausible cause for Pete's envy than Pete's success, hence the solution is 'he_m'.

Such reasoning is already challenging, however it is not always sufficient: there are WSs where both possible coreferences correspond to a world where the sentence makes sense — text structure and world knowledge are not sufficient for tackling Winograd Schemas.

One example for such a schema is the following.

$$\begin{array}{l}\text{[Sam's drawing]}_s \text{ was hung just above [Tina's } \emptyset]_t \text{ and}\\ \text{it}_{s,t?} \text{ did look much better with another one below it.}\end{array} \quad (5)$$

$$\begin{array}{l}\text{[Sam's drawing]}_s \text{ was hung just above [Tina's } \emptyset]_t \text{ and}\\ \text{it}_{s,t?} \text{ did look much better with another one above it.}\end{array} \quad (6)$$

(The '$\emptyset$' indicates an omission (ellipsis) of 'drawing'.)

In these sentences we can imagine a world where the drawing 'below it' or 'above it' is a third drawing, neither Sam's nor Tina's: then the pronoun can be either 'it_s' or 'it_t'.

Humans intuitively chose 'it_s' for (5) and 'it_t' for (6), intuitively because these readings of the sentences are much *more relevant* and *less far-fetched* than their alternatives.

In this work we focus on the disambiguation of such Winograd Schemas, schemas where we have to go beyond semantics and text structure because they are not sufficient for disambiguation.

We will approach this problem using Relevance Theory (Wilson and Sperber 2006), which measures relevance in

terms of *positive cognitive effect* – intuitively defined as the amount of inferences that are possible – minus the *processing effort* of achieving that effect. Relevance Theory is based on informal notions of human reasoning, hence integrating this theory into formal reasoning is not straightforward.

We propose a method for disambiguating WSs based on a formalization of criteria of Relevance Theory and present the following contributions.

- We formally describe a *knowledge graph* data structure which is a simplification of Roger Schank's psychological model of natural language understanding (Schank 1972).
- We formally describe a *reasoning process* that combines an input knowledge graph with background knowledge graphs, including constraints on certain intuitively necessary graph properties.
- Based on notions from Relevance Theory we propose a *relevance fitness function* on knowledge graphs.
- We report on *experiments* for *evaluating parameters* of that fitness function using Answer Set Programming.

Preliminaries

Winograd Schemas (WSs) were described by Winograd (1972), they are twin sentences with an ambiguous pronoun (see Hirst (1981) for a survey) and a *special word* such that exchanging this word with its *alternate word* changes the reference to another entity in the sentence. For example in the pair (1)/(2), the pronoun 'it' is ambigous between antecedent noun phrases 'a gap' and 'the wall', the special word is 'through' with its alternate word 'behind', the correct coreference is 'it_g' for (1) and 'it_w' for (2).

Levesque, Davis, and Morgenstern (2012) proposed the Winograd Schema Challenge as an alternative for the Turing Test, arguing that disambiguating WS requires true understanding of text on the one hand, and is more practical and more reasonable than the Turing Test on the other hand because (i) it does not require deception by the computer, (ii) does not require conversation, (iii) permits objective and automatic scoring, and (iv) requires deep reasoning over the input and the utilization of world knowledge.

They describe two important conditions on WSs used in such a challenge: (1) it shall not be possible to disambiguate the coreference by *selectional restrictions*, i.e., both antecedents must be of the same gender and number, moreover *straightforward statistical methods* shall not allow a clear disambiguation.[1] Moreover, (2) untrained human subjects shall identify the *correct* answer *without effort*.

The aim of the first condition is to require true text understanding to achieve success in the challenge, the second condition aims at a strong methodology with repeatable tests.

The original proposal of the challenge contains sentence pairs with a question, where the answer of the question asks for the referent of the pronoun. We will omit that question and instead check if the pronoun was resolved correctly.

Pragmatics is the linguistic field that analyses the effects of *context* on utterances. Note that'utterance' in linguistics denotes written, spoken, and signed natural language, similarly 'speaker' resp., 'hearer' denotes producers resp., consumers of natural language in all modalities. Moreover 'context' denotes linguistic, historical, cultural, and situative context.

From the perspective of WSs, pragmatics can provide useful insights about how the first part of a WS (the context) influences anaphora resolution in the second part (where the pronoun is located).

Relevance Theory (RT) is a popular theory within pragmatics, developed by Wilson and Sperber (2006). Their *Communicative Principle of Relevance* states that every utterance raises the expectation of its own optimal relevance, i.e., a hearer may interpret an utterance under the assumption that it is most relevant among all possible alternative utterances.

RT aims to 'explain in cognitively realistic terms what these expectations of relevance amount to, and how they might contribute to an empirically plausible account of comprehension.' This aim led to cognitive results that we will carry over to terms of knowledge representation, automated reasoning, and computation.

We will focus on three definitions of RT: relevance, processing effort, and positive cognitive effect.

Relevance of an utterance depends on two factors: other things being equal, (a) the greater the *processing effort* spent on interpreting an utterance, the lower its relevance, and (b) the greater the *positive cognitive effects* achieved by interpreting an utterance, the greater its relevance.

Processing effort is defined as the effort spent on perception, memory, and inference. Different contexts have different levels of accessibility (salience) for certain concepts such that processing effort for the same utterance can differ greatly between contexts.

Positive Cognitive Effect is 'a worthwhile difference to the individual's representation of the world'. Concretely it is defined wrt. the knowledge that can be concluded by a hearer using some mechanism of inference in human cognition.

According to RT, the most important cognitive effect is *contextual implication*: knowledge inferred from context C and utterance U that is not inferrable from C or U alone. Further positive cognitive effects are strengthening, revision, or abandonment of available *assumptions*.

Roger Schank's Knowledge Representation (1972) aims to provide a theory of human natural language understanding based on a graph formalism of concepts and dependencies that represent utterances and whole discourses.

A concept is either a nominal (e.g., 'book' or 'John'), an action (e.g., most verbs), or a modifier (e.g., adjectives). Dependencies can be of several types and directions, for example 'John $\leftrightarrow$ sleep' means that John sleeps. They can be annotated with temporal information, e.g., 'I $\leftrightarrow^p$ hit $\leftarrow^o$ John' means that John was hit by me (o indicates the object role and p the past).

Reasoning is described as looking up graph snippets from a library when encountering related input, a 'conceptual processor' that connects graph snippets according to conceptual rules, a 'syntactic processor' that ensures no syntactic constraints are violated. Later work by Schank (1980) makes several of these ideas more concrete (in particular memory

[1] As an example, if the antecedents are 'women' and 'pills' and the special words are 'pregnant' and 'carcinogenic', then the schema is too obvious to resolve, it is 'not Google-proof'.

organization) but is still far away from a formal account of how that reasoning is performed.

Related Work

Rahman and Ng (2012) proposed a machine-learning approach for tackling Winograd Schemas. They take into regard features such as narrative chains, FrameNet information, semantic compatibility, and polarity biases, and report a correctness result of 73% compared to 55% obtained via the state-of-the-art Stanford resolver (Lee et al. 2011). Some of their features are related to biases in language cognition.

Such biases are described by Kehler et al. (2008) who describe that pronoun interpretation is influenced by which coherence relations are likely to appear and which expectations about following entities are likely to occur. This is explained in more detail by Hartshorne (2013): e.g., a pronoun in a 'result' clause will most likely refer to the previous clauses' affected entity, while a pronoun in an 'explantory' clause most likely refers to entities that are causes.

As WSs are twin sentences, these biases should not make disambiguation easier. However, note that this is not true, if the word that evokes the bias is the special word itself (i.e., 'because' and 'although' should not be special words).

Relevance Formalizations. Gärdenfors (1978) discusses a logical account of relevance that formalizes whether some piece of logical evidence can influence the truth value of a sentence. Delgrande and Pelletier (1998) go a step further and define relevance as a property of sentences in relation to conditionals in conditional logic (which allows nonmonotonic inferences). They state that relevance in a logical formalism must be defined on the meta-level instead of within that formalism. We similarly apply our relevance fitness function on a meta-level to evaluate the reasoning process and its potential outcomes.

Understanding natural language requires taking assumptions and truly engaging in a process of *interpreting* a given linguistic input. This makes natural language so efficient but allows for complex misunderstandings. Relevance Theory, and the notion of relevance we pursue in this work, aims to pinpoint the *correct assumptions* a hearer has to make about a linguistic input such that the hearer obtains the interpretation intended by the speaker. (E.g., a flight attendant can use 'seat 3a' to refer to a passenger or to the purchase of a passenger at that seat.) Hence our notion of relevance is related to Gärdenfors, Delgrande and Pelletier's approaches but it is about human interpretation rather than about logical truth.

Hobbs et al. (1993) interpret natural language by abducing as few as possible assumptions to explain as much as possible observed input. The approach we introduce in the following can be considered to be abduction, but we evaluate relevance not only on the amount of abduced pieces of knowledge but in a more holistic way that considers global as well as local graph properties. Moreover our approach considers how well background knowledge integrates into the input during the reasoning process.

Textual Structure. As noted in the introduction, many Winograd Schemas can be solved by identifying expectations evoked by the textual structure of the input, and verifying if these expectations are violated by the semantic content of one or the other reading of the input.

One theory explaining such expectations is Rhetorical Structure Theory (RST) (Mann and Thompson 1988; Taboada and Mann 2006) which describes text organization as relations between parts of text, mostly between pairs of adjacent text spans. Such relations can be explicit in an utterance, e.g., the words 'if' or 'because', but they can be implicit, which creates the challenge of detecting them.

RST argues with similar principles as Relevance Theory, explaining textual structure as *providing relevant knowledge* that can be *integrated readily* into context and also *strengthening or justifying assumptions*. Both theories are likely to be necessary in a system that can disambiguate WSs.

Graph-Based Knowledge Representation. Important works related to our graph knowledge representation are knowledge bases and ontologies for natural language processing such as the Component Library (Barker, Porter, and Clark 2001), Generalized Upper Model (Bateman et al. 2010), and YAGO (Suchanek, Kasneci, and Weikum 2007). These and similar ontologies use formal logic (often Description Logics (Baader et al. 2003)) to categorize and relate linguistic terms, they are systematically engineered to cover a broad range of domains and applications. Contrary to that we use a lightweight graph framework to represent linguistic input and background knowledge, with the aim of studying the formalization of relevance. We hope that the results of this work can be applied also to these more systematic ontologies, as they have natural representations as graphs. As stated by Chaudhri, Dinesh, and Inclezan (2013), to be useful in natural language processing an ontology must be *linguistically motivated*.

Conceptual Blending (Pereira 2008) is an approach that is structurally similar to our knowledge graphs, however Blending has the goal to create *creativity* in AI systems. This is achieved by 'blending' knowledge graphs using associations of concepts that have similar properties, giving rise to 'metaphorical' and 'creative' computational reasoning.

Relevance in Conceptual Blending is formulated as a fitness *with respect to a reasoning goal*, for example the 'horse' and the 'bird' microtheory can be blended in different ways, and if we impose the goal 'flying' then a blend that is compatibly with flying is more relevant.

Knowledge Graphs and Reasoning

Schank aimed to provide a psychological model of how human reasoning works for understanding text. We next define a graph framework inspired by this model and formalize reasoning based on combining nodes of knowledge graphs without violating certain constraints of these graphs.

We simplify the framework to a form that is sufficient for our purposes. Most importantly we distinguish dependencies only by label, not by drawing direction or arrow style.

Knowledge Graphs

We define the knowledge graph data structure based on two vocabularies $\mathcal{C}_{cont}$ and $\mathcal{C}_{dep}$ which are the domains for labels of nodes and dependencies in our knowledge graph, resp.

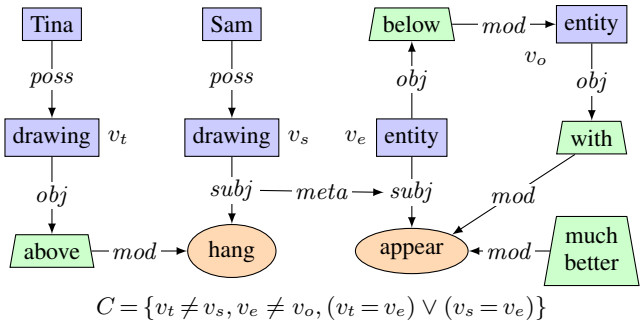

Figure 1: Knowledge graph structure corresponding to (5).

Example 1. *Figure 1 shows as an example the knowledge graph corresponding to input sentence* (5) *(we omit temporal information and interpret 'X's drawing' as 'the drawing owned by X'). Node labels are from the set* $\mathcal{C}_{cont} = \{$Tina, Sam, drawing, entity, hang, appear, above, below, with, much better$\}$. *Some intuitions about this graph: the* obj *dependency between* v_e *and 'below' denotes that* v_e *is the argument of 'below'; the 'below* v_e*' concept modifies* v_o *such that* v_o *denotes 'an entity that is below* v_e*'; moreover* v_e *has a* $subj$ *role in 'appear(ing)' which is modified by 'much better' and by 'with* v_o*'. The* $meta$ *dependency denotes an underspecified textual structure relation.*

In the following we implicitly extend this vocabulary as needed. The dependency vocabulary $\mathcal{C}_{dep} = \{subj, obj, obj2, mod, poss, at, meta\}$ is fixed.

We define an oracle function $compat : 2^{\mathcal{C}_{cont}} \to \{0, 1\}$ that evaluates whether a subset $C \subseteq \mathcal{C}_{cont}$ of the vocabulary contains mutually compatible content. Intuitively $compat$ abstracts from a (possibly intricate) reasoning in an ontological knowledge base that determines whether linguistically distinct concepts can denote the same world concept. For example, a drawing and an entity can be present in one concept node, while a drawing and Sam cannot: they must stay in separate concept nodes.

In Schank's graphs, dependencies occur between nodes and between dependencies, hence we reify dependencies: we represent a dependency as a graph node and two edges.

Definition 1. *A* knowledge graph structure $G = (V, E, l_{cont}, l_{type}, C)$ *on vocabularies* $\mathcal{C}_{cont}$ *and* $\mathcal{C}_{dep}$ *is a directed graph with nodes* V*, edges* $E \subseteq V \times V$*, node labelling functions* $l_{cont} : V \to 2^{\mathcal{C}_{cont} \cup \mathcal{C}_{dep}}$ *and* $l_{type} : V \to 2^{\{\mathbf{act}, \mathbf{obj}, \mathbf{mod}, \mathbf{dep}\}}$*, and a set of reasoning constraints* C *of two forms: inequalities between vertices and disjunctions of equalities between vertices.*

Figure 1 depicts a knowledge graph structure. Here and in the following we show nodes of type **act**, **obj**, and **mod**, as ellipses, rectangles, and trapezes, respectively. Dependencies (nodes of type **dep**) are shown without frame, and we omit arrows into such nodes unless they are $meta$-dependencies. Note in particular the $meta$ dependency between two $subj$ dependencies.

Given a set C of reasoning constraints we denote by $e(C)$ the set of disjunctions of equalities and by $n(C)$ the set of inequalities in C. In our example, $e(C) = \{(v_t = v_e) \vee (v_s = v_e)\}$ and $n(C) = \{v_t \neq v_s, v_e \neq v_o\}$.

We next impose constraints on knowledge graph structures to define knowledge graphs. The constraints ensure that knowledge graphs have certain properties, motivated methodologically (e.g., nodes have only one type, dependencies have only one incoming and one outgoing edge to a concept node) or linguistically (e.g., a limitation of object and possessor dependencies for concept nodes).

Equality constraints are inspired by constraints in Minimal Recursion Semantics (Copestake et al. 2005) which is a formalism for parsing natural language that allows ambiguous representations for unresolved pronouns as in WSs. Inequality constraints arise from syntactic principles of natural language, for example Principle B of Government and Binding (Haegeman 1994) states that a pronoun must not be bound within the same linguistic clause.

The following definition restricts each node $v \in V$ to have exactly one type $l_{type}(v)$. Using this restriction, we denote by $\mathbf{dep}(V) = \{v \in V \mid l_{type}(v) = \{\mathbf{dep}\}\}$ the set of *dependency nodes* and by $\mathbf{con}(V) = \{v \in V \mid l_{type}(v) \in \{\{\mathbf{act}\}, \{\mathbf{obj}\}, \{\mathbf{mod}\}\}\}$ the set of *concept nodes*.

Definition 2. *A* knowledge graph $G = (V, E, l_{cont}, l_{type}, C)$ *on vocabularies* $\mathcal{C}_{cont}$ *and* $\mathcal{C}_{dep}$ *is a knowledge graph structure such that*

(A) for every $v \in V$ *it holds that* $|l_{type}(v)| = 1$*;*
(B) for every $v \in \mathbf{dep}(V)$ *it holds that* $l_{cont}(v) \subseteq \mathcal{C}_{dep}$ *and* $|l_{cont}(v)| = 1$*;*
(C) for every $v \in \mathbf{con}(V)$ *it holds that* $l_{cont}(v) \subseteq \mathcal{C}_{cont}$ *and* $compat(l_{cont}(v)) = 1$*;*
(D) for $r \in \{obj, obj2, poss\}$ *and for every* $v \in V$ *it holds that* $|\{u \in V \mid (u, v) \in E \text{ and } l_{cont}(u) = \{t\}\}| \leq 1$.
(E) for every $v \in \mathbf{dep}(V)$ *it holds that* $|\{u \in \mathbf{con}(V) \mid (u, v) \in E\}| \leq 1$ *and* $|\{u \in \mathbf{con}(V) \mid (v, u) \in E\}| \leq 1$.
(F) for every $u \in \mathbf{con}(V)$*, it holds that there is at most one pair* (v, w)*,* $v \in \mathbf{dep}(V)$*,* $l_{cont}(v) = \{at\}$*,* $w \in \mathbf{con}(V)$ *such that* $\{(u, v), (v, w)\} \subseteq E$.
(G) there is no pair (u, v) *with* $u \in \mathbf{con}(V)$*,* $v \in \mathbf{dep}(V)$*, and* $\{(u, v), (v, u)\} \subseteq E$.
(H) for every pair $u, v \in \mathbf{con}(V)$*,* $u \neq v$ *it holds that* $|\{w \in V \mid \{(u, w), (w, v)\} \subseteq E\}| \leq 1$ *and* $|\{w \in V \mid \{(v, w), (w, u)\} \subseteq E\}| \leq 1$.

Note that Figure 1 is a knowledge graph.

These conditions become useful next where we define a reasoning task that combines nodes in knowledge graphs. (A) ensures single types for nodes. (B) ensures that dependency nodes have a singleton content from $\mathcal{C}_{dep}$. (C) ensures that concept nodes have as content a subset of the vocabulary that can refer to the same object or activity or modifier in the world. (D) ensures that a node can have only a single dependency from an object (obj) second object ($obj2$) and possessor ($poss$). This requirement is related to a linguistic principle known as 'theta criterion' which states that if a word assigns theta roles (types of arguments) then each role must be filled by a unique object (for example the word 'between' requires two arguments of type obj and $obj2$ for tangible or numerical entities). (E) ensures that a dependency node has no more than one incoming and no more than one

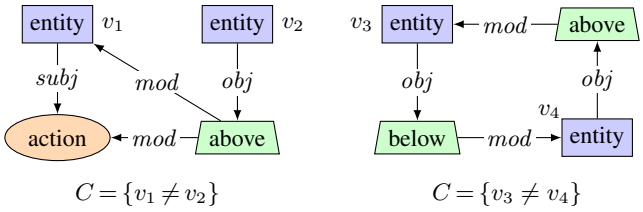

Figure 2: Background Knowledge Graphs related to (5).

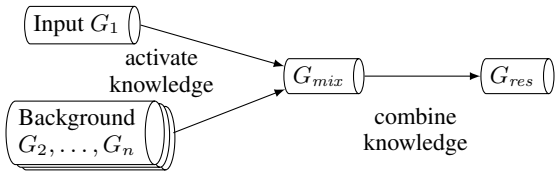

Figure 3: Reasoning with Knowledge Graphs

outgoing edge from (resp., to) a concept node. This causes dependency nodes to be like arcs between concept nodes (as visible in the figure) with the additional possibility of dependencies to dependencies. (F) ensures that a concept is at a single location only. (G) ensures that no concept depends on itself. (H) ensures that two concept nodes are never connected via more than one distinct dependency (in each direction). This avoids redundancies which will become important for realizing relevance in knowledge graphs.

From here on we leave vocabularies implicit.

Reasoning

We next define a notion of reasoning based on activating background knowledge graphs and combining them by collapsing subsets of nodes into single nodes. In the next section we define criteria for comparing all possible graphs resulting from such reasoning, yielding only those graphs that are most relevant wrt. principles of Relevance Theory.

Example 2. *Figure 2 depicts two knowledge graphs that are background knowledge related to our example. The left graph expresses that 'entity v_1 doing something above entity v_2' is interrelated with 'entity v_1 being above v_2'. The right graph expresses that 'entity v_3 being below entity v_4' is interrelated with 'entity v_4 being above v_3'. Note that these graphs does not imply a 'directionality', there are no 'conditions' or 'effects', just multiple dependency chains between concepts.*

Figure 3 gives an overview of our reasoning process: we activate knowledge in G_{mix} and then combine the activated graphs into result graph G_{res}.

Activating Knowledge. We start with a knowledge graph G_1 representing the input and a set $BG_{act} = \{G_2, \ldots, G_n\}$ of *activated knowledge graphs* from a background knowledge collection BG ($BG_{act} \subseteq BG$). For this work it was sufficient to activate knowledge $BG_{act} \subseteq BG$ according to matches of node contents that are more specific than 'entity' or 'action'. For a larger-scale system, methods that rely on salience and semantic distance should be used, for example Distributional Semantics (Padó and Lapata 2007).

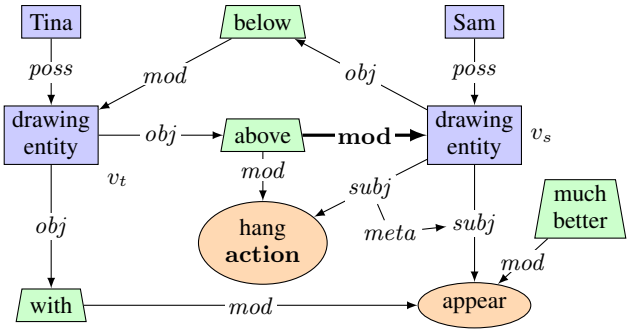

Figure 4: Reasoning: combining graphs from Figure 1 and 2.

Background knowledge is activated because it is somehow related to the input, however some of that knowledge might not be useful for reasoning. Hence we select a subset $BG_{use} \subseteq BG_{act}$ of *used knowledge graphs* from activated knowledge. The next section specifies how this selection is realized.

For combining knowledge we put all graphs in BG_{use} into one graph structure. For that we define a union operator: given two knowledge graphs G_1 and G_2 with disjoint sets of nodes, $G_i = (V_i, E_i, l_{cont_i}, l_{type_i}, C_i)$, $i \in \{1, 2\}$ we obtain $G_1 \cup G_2 = (V_1 \cup V_2, E_1 \cup E_2, l_{cont_1} \cup l_{cont_2}, l_{type_1} \cup l_{type_2}, C_1 \cup C_2)$. Due to disjoint sets of nodes the result is again a knowledge graph. If necessary, we can achieve disjointness by renaming nodes.

Combining Knowledge. We transform a knowledge graph by combining sets of nodes into single nodes. For that we use a function that maps nodes to other nodes.

Definition 3. *Given a knowledge graph structure $G = (V, E, l_{cont}, l_{type}, C)$, a* target function *$target : V \to V$ is a total function such that $target(v) = u$ with $v \neq u$ implies $target(u) = u$.*

A target function induces an equivalence relation over V: nodes that are mapped to themselves are representatives of the equivalence class and all other nodes map to such nodes. We use a function representation because it is more practical for our purposes.

The effect of applying a target function to a knowledge graph is defined next.

Definition 4. *Given a knowledge graph structure $G = (V, E, l_{cont}, l_{type}, C)$ and a target function $target$ wrt. G, we denote by $G[target]$ the* application of $target$ to G *and define $G[target] = (V', E', l_{cont}', l_{type}', C')$ with*

- $V' = \{target(v) \mid v \in V\}$,
- $E' = \{(target(u), target(v)) \mid (u, v) \in E\}$,
- $l_{cont}'(v) = \bigcup\{l_{cont}(u) \mid u \in V,\ target(u) = v\}$,
- $l_{type}'(v) = \bigcup\{l_{type}(u) \mid u \in V,\ target(u) = v\}$, *and*
- $C' = \{c$ *with vertices mapped using* $target \mid c \in C\}$.

The graph $G[target]$ is a knowledge graph structure, however it might not be a knowledge graph (e.g., a vertex might obtain two types). We next restrict $target$ such that

$G[target]$ is again a knowledge graph structure, and define the effect of reasoning constraints.

Reasoning constraints enforce structural properties of knowledge graphs: inequality constraints ensure that two nodes remain distinct nodes during reasoning, while equality constraints ensure that certain nodes are collapsed with others. In our example sentence (5) the word 'another' causes the constraint $v_e \neq v_o$ in Figure 1 while the word 'it' must bind to a predecessor phrase which can either be 'Sam's drawing' or 'Tina's [drawing]', hence we have the constraint $(v_t = v_e) \vee (v_s = v_e)$.

Definition 5. *Given a knowledge graph structure $G = (V, E, l_{cont}, l_{type}, C)$, a target function $target$ wrt. G is a* reasoning function *if*

(a) $G[target]$ is a knowledge graph,
(b) for each inequality constraint $(u \neq v) \in n(C)$ it holds that $target(u) \neq target(v)$, and
(c) for each equality constraint $c \in e(C)$ there is an equality $(u = v)$ in c such that $target(u) = target(v)$.

As an example consider Figure 4 where we combine knowledge graphs. If we disregard the bold **mod** edge and **action** concept then the graph is just a reorganized input graph (Figure 1) with $target(v_o) = v_t$ and $target(v_e) = v_s$. Combining the left graph in Figure 2 with the input graph is possible by combining 'action' with 'hang', and other nodes with nodes containing the same content, which leads to an additional dependency mod between 'above' and node v_s. Now we can see that the right graph from Figure 2 is already contained in the resulting graph, precisely we can combine v_3 with v_s and v_4 with v_t and then nodes containing 'below' and 'above', and this generates no additional edges or nodes. Note that these operations do not violate reasoning constraints, and that in the resulting set of constraints C' we still have $v_t \neq v_s$ but we omit the equivalence constraint because it is satisfied as v_s and v_e becoming the same node.

Observe that the result graph depicted in Figure 4 is a correct solution to the Winograd Schema (5) because in that schema 'it' must refer to Sam's drawing and the graph identifies v_e with v_s.

Finally we formally define a reasoning process based on which we will next formulate criteria of relevance.

Definition 6. *Given input graph G_1 and background knowledge BG, our reasoning process deterministically obtains BG_{act}, then chooses BG_{use} and $target$, and thereby obtain the result graph $G_{res} = G_{mix}[target]$ where*

$$G_{mix} = \bigcup \big(\{G_1\} \cup BG_{use}\big).$$

Relevance (Theory) in Knowledge Graphs

The previous section defined a reasoning process that leaves two important points open: (i) how to select BG_{act} and (ii) how to select $target$. In this section we use concepts from Relevance Theory to *assigning a relevance fitness value* to a reasoning process. This fitness value induces a preference between possible choices for BG_{use} and $target$.

By comparing fitness values wrt. BG_{act} and $target$ we can find the *most relevant* reasoning outcomes for a given input. This approach corresponds well with RT, which states that relevance cannot be measured in absolute terms, but it is always *in comparison with alternatives*.

RT originally is a theory of human cognition of natural language. We next justify and sketch possible formalizations of important principles from RT, then we propose a fitness function based on these ideas.

Nodes in G_{res} that originate from at least one node in the input graph G_1 we call *input nodes* (V_{res}^{in}), while nodes that originate only from nodes in background knowledge graphs we call *background nodes* (V_{res}^{bg}). In our example in Figure 4, the only background node is the bold **mod** dependency node, all other nodes are input nodes because the remaining (seven) nodes from background knowledge are combined with input nodes.

We consider the following formalizations of relevance.

• The amount of nodes from background knowledge that can been combined with input nodes indicates how well background knowledge integrates with the input. If all nodes from a background knowledge graph are combined with input nodes, then that background graph *supports existing knowledge* or *strengthens existing assumptions*. Both is defined as positive cognitive effect in RT.

• Background nodes can form *contextual implications* in the graph, also contributing to positive cognitive effect: they can connect parts of the input graph (as in Figure 4) that are not connected originally. Such connections are like inferences using background and input graph knowledge.

• Background nodes can be close to input nodes or far away in the graph (in terms of the shortest path to some input node). The former nodes are closely related to input, while the latter are more 'far-fetched', they can be seen as *additional assumptions*.

Assuming additional concepts can be necessary to make sense of an input, however this imposes additional *processing effort* and therefore reduces relevance relative to alternatives. In our example, we can combine the right graph in Figure 2 with Figure 1 in a more far-fetched manner: instead of combining v_3 with v_s and v_4 with v_t we could combine v_3 with v_t and not combine v_4 with any input node. In that case our reasoning process makes the *assumption* that there is a third object that is below Tina's drawing.

• Another effect of reasoning with background knowledge is that it can connect nodes from the input graph and thus causes the overall graph to become more connected. To measure how well the result of reasoning matches expectations in background knowledge, and how well-integrated the result of reasoning is, it can therefore be useful to measure the *radius* of the resulting knowledge graph.

The graph radius is the smallest value r such that from some node of the graph we can reach all other nodes traversing less than r edges. Our input graph in Figure 1 has radius 7 (from the left $subj$ node), while the result graph in Figure 4 has radius 5 (from the bold mod node). Recall that dependencies are nodes, so from 'Tina' to 'meta' we traverse 8 edges.

Based on our definition of reasoning process, we next define a fitness function with constant parameters C_{in}^{com}, C_{bg}, C_{bg}^{com}, C_{bg}^{dis}, C^{om}, and C^{rad}.

Definition 7. *Given G_1, BG, BG_{use}, and target as in Definition 5, we define fitness function f_{rel} as*

$$f_{rel}(G_1, BG, BG_{use}, target) = C_{in}^{com} \cdot \sum_{n \in V_{res}^{in}} com(n) + C_{bg} \cdot |V_{res}^{bg}| + C_{bg}^{com} \cdot \sum_{n \in V_{res}^{bg}} com(n) + C_{bg}^{dis} \cdot \sum_{n \in V_{res}^{bg}} dis(n) + C^{om} \cdot |BG_{act} \setminus BG_{use}| + C^{rad} \cdot rad(G_{res})$$

where

$$\begin{aligned} V_{res}^{in} &= \{target(v) \mid v \in V_1\}, \\ V_{res}^{bg} &= V_{res} \setminus V_{res}^{in}, \\ com(n) &= |\{v \in V_{res} \mid target(n) = target(v)\}|, \\ dis(n) &= \textit{the length of the shortest path from } n \\ &\quad \textit{to some node } n' \in V_{res}^{in}, \textit{ and} \\ rad(G_{res}) &= \textit{the radius of graph } G_{res}. \end{aligned}$$

Informally, V_{res}^{in} denotes input nodes, V_{res}^{bg} denotes nodes background nodes, $com(n)$ denotes the number of nodes combined to form node $n \in V_{res}$, and $dis(n)$ denotes the distance of node $n \in V_{res}^{bg}$ to the nearest node $n' \in V_{res}^{in}$.

This fitness function contains all points stated informally before, with (so far) unspecified constant factors as parameters. C_{in}^{com} allows to measure how many nodes were combined to yield input nodes. For background nodes, C_{bg} gives uniform weight to each one, C_{bg}^{com} gives weight proportional to the amount of combined nodes, and C_{bg}^{dis} gives weight proportional to the distance from input nodes. Moreover C^{om} allows for giving a weight to each activated background knowledge graph that was not used, and C^{rad} gives a weight to the radius of the result graph G_{res}.

Note that C_{in}^{com}, C_{bg}, and C_{bg}^{com}can be computed locally for each node of G_{res}, while C_{bg}^{dis} requires consideration of more than one node, C^{rad} is about a property of the whole graph, and C^{om} is not about the graph but about which background knowledge is used to build G_{res}. These locality properties can have significant influence on implementability and computational efficiency of an optimization procedure that computes graphs with optimal fitness. In this work we consider only small inputs to show qualitative applicability of the fitness function, therefore we will only briefly touch on matters of computational efficiency.

Experimental Evaluation

We designed our relevance fitness function based on principles of RT, moreover we know that disambiguation of certain WSs can be explained in terms of relevance.

In this section we perform experiments on such schemas in order to show which parameters lead to a correct disambiguation, and how well our fitness function performs in general, i.e., whether it captures the requirements on disambiguating our example schemas.

For our empirical experiments we selected Winograd Schemas numbered 2, 5, 15, and 17 from (Levesque, Davis, and Morgenstern 2012), encoded them in both of their variants as input graphs, and encoded some relevant background knowledge (including Figure 2). We repeat these schemas as (7)+(8), (5)+(6), (9)+(10), and (11)+(12), respectively.

[Tom]$_t$ threw his schoolbag down to [Ray]$_r$ after [he]$_{t,r?}$ reached the top of the stairs. (7)

[Tom]$_t$ threw his schoolbag down to [Ray]$_r$ after [he]$_{t,r?}$ reached the bottom of the stairs. (8)

The [drain]$_d$ is clogged with [hair]$_h$. [It]$_{d,h?}$ has to be removed. (9)

[The drain]$_d$ is clogged with [hair]$_h$. [It]$_{d,h?}$ has to be cleaned. (10)

There is [a pillar]$_p$ between me and [the stage]$_s$, and I can't see [it]$_{p,s?}$ (11)

There is [a pillar]$_p$ between me and [the stage]$_s$, and I can't see around [it]$_{p,s?}$. (12)

We chose these schemas because they do not violate text structure or world knowledge in their wrong reading, therefore we think Relevance Theory is a major ingredient in solving them.[2]

For each schema we computed the reasoning outcome(s) that were ranked as most relevant by the fitness function f_{rel}. We measured the number of result graphs with correct disambiguation (i.e., graphs where the node corresponding to the ambiguous pronoun is combined with the node of the input that corresponds to the correct answer) versus the number of result graphs with an incorrect disambiguation.

We performed this experiment on the following parameter combinations.

- $C_{in}^{com} \in \{0, 1, 2, 3\}$ varies the importance of combining background knowledge with input nodes.
- $(C_{bg}, C_{bg}^{com}, C_{bg}^{dis}) \in \{(0,0,0), (-1,0,0), (0,0,-1), (0,1,-2), (0,1,-1), (0,2,-1)\}$ varies the importance of several features of background nodes. In particular (0,0,0) omits any cost or benefit, (-1,0,0) gives a uniform cost to every background node, (0,0,-1) imposes a cost corresponding to distance from the nearest input node, (0,1,-1) additionally gives a benefit for combining with other nodes, finally (0,1,-2) and (0,2,-1) vary the ratio between cost and benefit of distance from input nodes and combination with background knowledge.
- $C^{om} \in \{0, -10, -20, -30\}$ varies the cost of not using an activated background knowledge graph.
- $C^{rad} \in \{0, -5, -10, -15\}$ gives cost to the graph radius.

These parameter ranges were established from estimates of parameter importance and in preliminary experiments. The seemingly natural setting $(C_{bg}, C_{bg}^{com}, C_{bg}^{dis}) = (0, 1, 0)$ was excluded because it facilitates the use of background knowledge that is neither combined with other background knowledge nor with input knowledge.

The above parameter combinations yield 384 overall parameter combinations and we have 8 sentences (2 for each schema), yielding 3072 overall experimental runs.

We encoded the reasoning process in Answer Set Programming (ASP) (Lifschitz 2008; Brewka, Eiter, and

[2]There can be two stairs, a plumber might remove a clogged drain, and there can be an object blocking the view to the pillar.

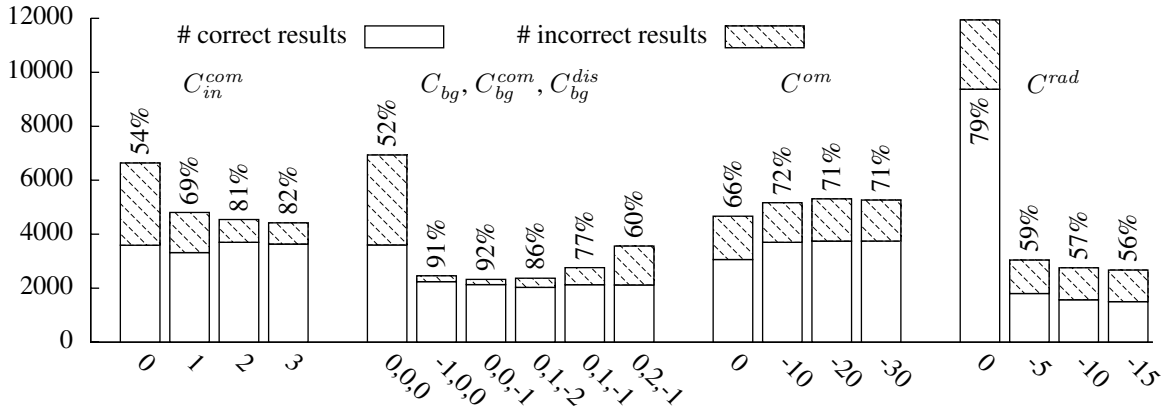

Figure 5: Experimental results (for each parameter group we accumulated over all combinations of remaining parameters).

Truszczynski 2011) which is a logic programming formalism suitable for representing knowledge and guess-and-check computations with integer optimization. Encodings, graph visualizations, and results are available online.[3]

In our experiments we used the solver clingo 4.2.1 (Gebser, Kaufmann, and Schaub 2013) with a timeout of 1 hour for finding the optimal value and 1 hour for enumerating a maximum of 25 graphs of optimum cost. Note that our encodings are not tuned for good performance, hence some parameter combinations and inputs cause timeouts and we only consider results from runs without timeouts. Such timeouts happened only for inputs (7) and (8), due to a high number of activatable background knowledge graphs. To gain confidence in our results, we checked that all trends we report in the following are similar if we use a timeout of only 20 min, hence timeouts do not invalidate the following discussion.

Results

Figure 5 shows results for each of the four parameter groups described previously, accumulated over all possible combinations of the other three parameter groups. We display the number of enumerated reasoning graphs, split into correctly and incorrectly disambiguated graphs, we also show the percentage of correct disambiguations.

We aim for a relevance fitness function that has the following properties: (i) firstly it gives higher fitness to graphs that contain the correct disambiguation, and (ii) it select as few as possible best results. The benefit of (i) is obvious, while (ii) is desireable for reasons of relevance: a low total number of results means that few assumptions needed to be taken for achieving these results, and few assumptions are necessary for subsequent reasoning.

From Figure 5 we can observe the following.

- Increasing C_{in}^{com} improves the correctness and reduces the number of result graphs.
- Giving weight to the second group of parameters $(C_{bg}, C_{bg}^{com}, C_{bg}^{dis})$ is important for correctness and for reducing the number of result graphs. The best correctness results are achieved with (-1,0,0), (0,0,-1), and (0,1,-2).
- A nonzero C^{om} increases correctness and increases the number of results (due to encouraging usage of background graphs).
- Giving weight to C^{rad} reduces correctness from 79% to 59% and drastically reduces the number of results.

These accumulated results give an overview of the impact of our fitness function constants. We next look into details.

Table 1 shows the best and worst individual parameter combinations. Each row is the result of 8 runs (one for each sentence of each schema).

In the upper part of the table we first show the 11 best combinations, they yield 100% correct answers and only 10 answers for 8 inputs. We can observe that C_{in}^{com} and C^{rad} are never 0, furthermore the combinations (0,0,0) and (0,2,-1) do not appear in the second column, which indicates that these parameter values are not favorable for correctness. C^{om} seems to be least important. The table row with 11 correct answers and 100% correctness actually contains 30 different parameter configurations, these combinations are not shown for space reasons, but we note that these combinations contain neither zero constants for C_{in}^{com}, C^{om}, and C^{rad}, nor combinations (0,0,0) or (0,2,-1) for the other constants. This suggests that all possible parameter values except (0,0,0) and (0,2,-1) and zero values contribute to successful disambiguation.

In the lower part of the table we show the 8 worst settings. (C_{bg},C_{bg}^{com},C_{bg}^{dis}) is either (0,0,0) which gives no weight to background nodes, or (0,1,-1), which gives cost to distance from input nodes but also gives the same amount of benefit for combination with other background knowledge nodes. C_{in}^{com} is zero in all but configuration. C^{om} and C^{rad} vary across their possible values. The total number of solutions (correct plus incorrect) is higher in the lower part of the table, except for two rows with 7 incorrect and 4 correct solutions: in these rows the parameters make graph radius the dominating cost and optimal fitness is achieved by not using any background knowledge (which is undesired and also leads to bad correctness).

In summary we see that all coefficients of f_{rel} are important, and many parameter values and combinations achieve correct disambiguation with few reasoning result. On the

[3] www.peterschueller.com/winograd/kr14/

C_{in}^{com}	(C_{bg},C_{bg}^{com},C_{bg}^{dis})	C^{om}	C^{rad}	# incorr.	# corr.	% corr.
3	(0,1,-2)	0	-5	0	10	100
3	(0,1,-2)	0	-10	0	10	100
3	(0,1,-2)	-10	-5	0	10	100
3	(0,1,-2)	-10	-10	0	10	100
3	(0,0,-1)	0	-5	0	10	100
1	(0,0,-1)	-10	-5	0	10	100
3	(0,0,-1)	-10	-5	0	10	100
3	(0,0,-1)	-10	-10	0	10	100
2	(-1,0,0)	0	-5	0	10	100
3	(-1,0,0)	0	-5	0	10	100
3	(0,1,-1)	0	-5	0	10	100
— 30 configurations (see text) —				0	11	100
.	.	.	.	...	...	...
0	(0,1,-1)	0	-15	7	4	36
0	(0,1,-1)	0	-10	7	4	36
1	(0,0,0)	-20	0	102	58	36
0	(0,0,0)	0	-15	100	50	33
0	(0,0,0)	0	-5	100	50	33
0	(0,0,0)	0	-10	100	50	33
0	(0,1,-1)	-10	0	114	46	29
0	(0,0,0)	-30	-5	154	24	14

Table 1: Best and worst parameter settings (8 runs per row).

negative side we see that zero values are unfavorable, and that the benefit of (0,1,-1) depends on other parameters.

Conclusion and Future Work

In order to formalize important concepts from Relevance Theory we introduced a formal simplification of Roger Schank's framework for knowledge graphs. We started from this framework because it promised an easy formulation of principles from RT as properties of graphs. Moreover, our knowledge graphs have similar properties to other graphs used for representing knowledge.

Therefore we hope that our work can inspire applications of Relevance Theory also in other graph-based reasoning formalisms.

Negative Results. Preliminary experiments showed that certain graph parameters are not useful: minimizing the number of *bridges* (edges whose removal disconnects the graph) does not yield promising results because the input graph might already be without any bridges. Also, maximizing the number of dependencies per node encourages buildup of additional dependencies between input nodes (think: contextual implications) but this measure can introduce redundancies in the graph that do not violate basic graph conditions and certainly do not follow the intended notion of relevance.

Possible Extensions. An important possible extension of our approach are graphs that act as *constraints*, i.e., they invalidate a reasoning result if a specific part of them *can* match the result graph. Preliminary experiments show that adding only one background knowledge graph with such capabilities increases the correctness of many parameter combinations tested in this work. Due to prohibitively high computation times we conducted only preliminary experiments.

Possible extensions of our relevance fitness function are the Cheeger constant (a graph parameter that measures *bottlenecks* in a graph; it is mainly used in research about social or computer networks) and a variable cost for unused activated graphs, depending on such graphs' size.

Winograd Schemas contain short sentences and they are meant to be interpreted without additional context. Therefore wrt. processing effort we mainly considered effort of assuming additional entities, a characterization that is shared by RT and by Hobbs et al. (1993). Future work should also consider salience of concepts and semantic distance between concepts as factors that contribute to processing effort.

Obtaining Knowledge Graphs. We here considered only knowledge graphs and abstracted from the process of *parsing natural language* to create knowledge graphs — certainly a challenging task by itself. Most existing parsing systems resolve ambiguities during parsing, a notable exception is Minimal Recursion Semantics (Copestake et al. 2005) which inspired equality constraints in this work. For obtaining and processing knowledge graphs several related works exist, e.g., the Boxer system (Bos 2008) which is based on Discourse Representation Theory, work by Baral and Dzifcak (2012) who interpret natural language and inferring unknown annotations using inverse lambda calculus, and the approaches of Chen et al. (2013) who combine existing open knowledge bases in a system for human-robot interaction.

Probabilistic approaches are not included in this work, in a general and robust system such methods will be necessary, and metrics relevant to probabilistic NLP will become relevant for evaluating such a system, for example MUC (Message Understanding Conference) metrics.

Computational Efficiency. We largely disregarded the efficiency of reasoning in our experiments and discussion. Due to the high complexity class of our reasoning method which is a typical guess & check & optimize task, reasoning efficiency will certainly become an important topic in practice. Several possibilities for tuning performance exist, the most obvious being an optimization of the ASP encoding, using general principles as described, e.g., by Balduccini and Lierler (2012). Moreover there is the possibility of using a hybrid reasoning formalism such as HEX-programs (Eiter et al. 2005) that combine ASP with domain-specific C++ code (e.g., for graph processing). Finally it is possible to realize the reasoning method from this paper in a formalism that is independent from ASP.

Incremental Progress. While our experiments indicate that certain parameter combinations lead to correct disambiguation, they are limited because they have been performed on a small set of examples. Levesque, Davis, and Morgenstern (2012) pointed out that *incremental progress* is possible in the WSC, by creating libraries of questions that can be solved by children, up to questions solvable by university-educated people. They also propose using a limited vocabulary, which will make testing more objective, and avoid solutions that are targeted to certain sentences.

For this paper we created a few background knowledge graphs from which several become activated and used in multiple of our test sentences. Nevertheless we are convinced that for achieving truly general results, it will be *nec-*

essary to create larger-scale testsuites on standard vocabularies. (Otherwise background knowledge used in experiments might just be tuned to achieve the right experimental outcome, making the experimental results meaningless.)

Creating such a corpus of Winograd Schema on a limited vocabulary contains several research challenges such as vocabulary selection, building the schemas, and evaluating empirically whether each schema is correctly disambiguated by a significant majority of human subjects.

Acknowledgements

We thank the anonymous reviewers for their constructive feedback, and Roland Kaminski for finishing clingo 4.2 at exactly the right time and for fast reactions to bug reports.

References

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P., eds. 2003. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press.

Balduccini, M., and Lierler, Y. 2012. Practical and Methodological Aspects of the Use of Cutting-Edge ASP Tools. In *International Symposium on Practical Aspects of Declarative Languages (PADL)*, 78–92.

Baral, C., and Dzifcak, J. 2012. Solving Puzzles Described in English by Automated Translation to Answer Set Programming and Learning How to Do that Translation. In *International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 573–577.

Barker, K.; Porter, B.; and Clark, P. 2001. A library of generic concepts for composing knowledge bases. In *International Conference on Knowledge Kapture (K-CAP)*, 14.

Bateman, J. a.; Hois, J.; Ross, R.; and Tenbrink, T. 2010. A linguistic ontology of space for natural language processing. *Artificial Intelligence* 174(14):1027–1071.

Bos, J. 2008. Wide-coverage semantic analysis with Boxer. In *Conference on Semantics in Text Processing*, 277–286.

Brewka, G.; Eiter, T.; and Truszczynski, M. 2011. Answer set programming at a glance. *Comm ACM* 54(12).

Chaudhri, V.; Dinesh, N.; and Inclezan, D. 2013. Three lessons in creating a knowledge base to enable reasoning, explanation and dialog. In *Workshop on Natural Language Processing and Automated Reasoning (NLPAR)*, 7–26.

Chen, X.; Xie, J.; Ji, J.; and Sui, Z. 2013. Toward Open Knowledge Enabling for Human- Robot Interaction. *Journal of Human-Robot Interaction* 1(2):100–117.

Copestake, A.; Flickinger, D.; Pollard, C.; and Sag, I. A. 2005. Minimal Recursion Semantics: An Introduction. *Research on Language and Computation* 3(2-3):281–332.

Delgrande, J. P., and Pelletier, J. 1998. A Formal Analysis of Relevance. *Erkenntnis* 49(2):137–173.

Eiter, T.; Ianni, G.; Schindlauer, R.; and Tompits, H. 2005. A Uniform Integration of Higher-Order Reasoning and External Evaluations in Answer-Set Programming. In *International Joint Conference on Artificial Intelligence*, 90–96.

Gärdenfors, P. 1978. On the Logic of Relevance. *Synthese* 37:351–367.

Gebser, M.; Kaufmann, B.; and Schaub, T. 2013. Advanced Conflict-Driven Disjunctive Answer Set Solving. In *International Joint Conference on Artificial Intelligence*, 912–918.

Haegeman, L. 1994. *Introduction to Government and Binding Theory*. Blackwell.

Hartshorne, J. K. 2013. What is implicit causality? *Language, Cognition and Neuroscience*.

Hirst, G. 1981. *Anaphora in Natural Language Understanding: A Survey*. Springer-Verlag.

Hobbs, J. R.; Stickel, M.; Martin, P.; and Edwards, D. 1993. Interpretation as abduction. *Artif Intell* 63(1-2):69–142.

Kehler, A.; Kertz, L.; Rohde, H.; and Elman, J. 2008. Coherence and coreference revisited. *Journal of Semantics* 25(1):1–44.

Lee, H.; Peirsman, Y.; Chang, A.; Chambers, N.; Surdeanu, M.; and Jurafsky, D. 2011. Stanfords Multi-Pass Sieve Coreference Resolution System at the CoNLL-2011 Shared Task. In *CoNLL: Shared Task*, 28–34.

Levesque, H.; Davis, E.; and Morgenstern, L. 2012. The Winograd Schema Challenge. In *International Conference on Principles of Knowledge Representation and Reasoning*.

Lifschitz, V. 2008. What Is Answer Set Programming? In *AAAI Conference on Artificial Intelligence*, 1594–1597.

Mann, W. C., and Thompson, S. A. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text* 8(3):243–281.

Marcus, G. 2013. Why can't my computer understand me? *The New Yorker*. August 18.

Padó, S., and Lapata, M. 2007. Dependency-Based Construction of Semantic Space Models. *Computational Linguistics* 33(2):161–199.

Pereira, F. C. 2008. *Creativity and AI: A Conceptual Blending approach*. Walter de Gruyter.

Rahman, A., and Ng, V. 2012. Resolving complex cases of definite pronouns: The Winograd Schema Challenge. In *Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNPL)*, 777–789.

Schank, R. 1972. Conceptual dependency: A theory of natural language understanding. *Cognitive Psychol* 3:552–631.

Schank, R. C. 1980. Language and Memory. *Cognitive Science* 4(3):243–284.

Suchanek, F. M.; Kasneci, G.; and Weikum, G. 2007. YAGO: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In *International Conference on World Wide Web (WWW)*, 697–706.

Taboada, M., and Mann, W. C. 2006. Rhetorical Structure Theory: looking back and moving ahead. *Discourse Studies* 8(3):423–459.

Wilson, D., and Sperber, D. 2006. Relevance theory. In Horn, L., and Ward, G., eds., *The Handbook of Pragmatics*. Blackwell. 607–632.

Winograd, T. 1972. *Understanding Natural Language*. Academic Press.