FrameBase: Representing N-ary Relations using Semantic Frames

Jacobo Rouces[1], Gerard de Melo[2], and Katja Hose[1]

[1] Aalborg University, Denmark
jrg@es.aau.dk, khose@cs.aau.dk
[2] Tsinghua University, China
gdm@demelo.org

Abstract. Large-scale knowledge graphs such as those in the Linked Data cloud are typically represented as subject-predicate-object triples. However, many facts about the world involve more than two entities. While n-ary relations can be converted to triples in a number of ways, unfortunately, the structurally different choices made in different knowledge sources significantly impede our ability to connect them. They also make it impossible to query the data concisely and without prior knowledge of each individual source. We present FrameBase, a wide-coverage knowledge-base schema that uses linguistic frames to seamlessly represent and query n-ary relations from other knowledge bases, at different levels of granularity connected by logical entailment. It also opens possibilities to draw on natural language processing techniques for querying and data mining.

1 Introduction

Over the past few years, large-scale knowledge bases (KBs) have grown to play an important role on the Web. Many institutions rely on Linked Data principles to publish their data using Semantic Web standards [2]. These KBs are mostly based on simple subject-predicate-object (SPO) triples, as defined by the RDF model [15]. Such triples are convenient to process and can be visualized as entity networks with labeled edges.

Whereas triple representations work straightforwardly for relations involving two entities, many interesting facts relate more than just two participants – a problem that has gained renewed attention in several recent papers [13, 22] as well as in the current W3C proposal to add roles to schema.org [1]. For a birth event, for instance, one may wish to capture not just the time but also the location and parents. For an actress starring in a movie, the name of the portrayed character may be relevant. Such facts naturally correspond to n-ary relations. In order to capture them as triples, several different representation schemes have been proposed. Table 1 shows some possibilities of expressing that an entity John was married in 1964, some of which also include additional information such as the name of the bride. We will discuss these representations in more detail later in Sect. 2.

As the example shows, this sort of semantic heterogeneity leads to significant data integration challenges. One KB might use a simple binary property between two entities, whereas another may instead choose a more complex representation that accommodates additional arguments. The representations can easily be so at odds with each other that no particular mapping between entities could bridge the differences. There are entities

at each side that have no counterpart at the other. This leads to several challenging problems:

1. When **linking data**, there are currently no mechanisms to connect KBs with different modeling choices. Predicates exist to link equivalent classes, instances, or properties, but not for connecting the different patterns, as explained above. Existing work on ontology and KB alignment [3] is limited to finding aliases.
2. When **querying**, the query must be built in a way that fits the particular modeling choices made for the respective KB. Otherwise, the recall may be as low as zero [26]. Even worse, when we don't have a single coherent KB but a set of different KBs, there is no simple query (as could be formulated on a single given schema) that will have a high recall across all KBs.
3. When **natural language interfaces** to KBs are queried, state-of-the-art systems typically attempt to map verbs and predicate phrases to RDF predicates [33]. This approach, however, cannot be applied when the KB fails to provide a compatible binary relation.

Direct Binary Relation

```
John   marriedOnDate  1964 .
```

RDF Reification

```
John  marries         Mary .
s     type            Statement .
s     subject         John .
s     property        marries .
s     object          Mary .
s     time            1964 .
```

Subproperties

```
p     subPropertyOf   Marriage .
John  p               Mary .
p     time            1964 .
```

Neo-Davidsonian (Specific Roles)

```
e  type     Marriage .
e  groom    John .
e  bride    Mary .
e  time     1964 .
```

Neo-Davidsonian (General Roles)

```
e  type     Marriage .
e  agent    John .
e  agent    Mary .
e  time     1964 .
```

Table 1: Triple Representations of n-ary Relations

FrameBase. To address these problems, we have created FrameBase, a broad-coverage schema that can homogeneously integrate other KBs and has strong connections to natural language. It overcomes the above-mentioned forms of heterogeneity – by sticking to a specific modeling choice general enough to subsume the others (neo-Davidsonian representation) – together with a large vocabulary for events and roles. This vocabulary is reusable and based on an extensible hierarchy. We also develop a mechanism to convert back and forth between the new representation and direct binary relations, using a vocabulary of binary relations automatically generated from linguistic annotations. These are more concise and can be used when only two arguments are relevant.

This paper is structured as follows. After analyzing the state of the art in Sect. 2, an overview of FrameBase is given in Sect. 3. Section 4 explains how we construct the FrameBase schema, while Sect. 5 presents our representation conversion mechanism. Section 6 provides a qualitative evaluation, and Sect. 7 concludes the paper with an outlook to future work.

	All triples	Core	Linking event	Reasoning
RDF Reification	$(n+4)k$	$(n+3)k$	$+k(k-1)$	One definite clause
Subproperties	$(n+2)k$	$(n+1)k$	$+k(k-1)$	RDFS
Neo-Davidsonian	$1+n+k$	$1+n$	$+0$	Several def. clauses

Table 2: Triple Overhead. n *is the number of participants in an event, and* k *the number of pairs that are relevant to be linked by direct binary relations. The first column indicates the total number of triples that can be materialized. The second column excludes direct binary relationships, which can be inferred unambiguously by the inference system in the last column. In the case of RDF reification, this inference could be accomplished by a rule creating the triple from its reification triples. In the case of neo-Davidsonian representation, we use rules of a different form (described later in Sect. 5). In both cases, each rule is a definite clause, i.e. a disjunction of logical atoms with only one negated, which is the consequent when the clause is written as an implication. The third column indicates the number of triples needed to connect entities that represent the same event, which is a phenomenon that arises when using RDF reification or subproperties.*

2 State of the Art

In this section, we review related work and conduct a thorough analysis of existing approaches for modeling n-ary relations, which are synthesized in Table 1. In Table 2, we provide a detailed comparison of their space efficiency, which has consequences with regards to their applicability for large-scale KBs.

2.1 Direct Binary Relations

A common way to represent n-ary facts is to simply decompose them directly into binary relations between two participants [8]. But in doing so, important information may be lost. For instance, given a triple with property `wasMarriedOnDate` and two triples with `gotMarriedTo`, we cannot be sure to which marriage the given time span applies.

2.2 RDF Reification

The RDF standard proposes RDF reification [15], which introduces a new identifier (IRI) for a statement and then describes the original RDF statement using three new triples with `subject`, `predicate`, and `object` properties. Subsequently, arbitrary properties of the statement can be captured by adding further triples about it.

In the different versions of YAGO [16], RDF reification is used to attach additional information to the event represented by the original RDF triple (evoked by its property) – as in the *RDF Reification* example in Table 1. This has the advantage that both the original triple as well as the reified triple can be present in the KB and queries that do not require the additional information can still use the original binary relation directly. However, this also has several drawbacks:

– Formally, the event represented by a triple and the triple as a statement are different entities with different properties. For instance, an institution may endorse the triple as a statement without endorsing the marriage. Using RDF reification, both are represented by the same RDF resource identifier, which conceptually is meant to be unambiguous. This is a potential source of confusion and inconsistency.

– The number of triples increases by a factor of 4. For each triple `S P O`, one has to add `T a rdf:Statement`, `T rdf:subject S`, `T rdf:predicate P`, and `T rdf:object O`. These do not add any new information themselves but are merely a prerequisite for then being able to extend the original binary relation to an n-ary relation by subsequently adding more triples with `T` as subject.
– The advantage of being able to include the original non-reified triple only applies for the primary binary relation, and not for the other $\frac{n(n-1)}{2} - 1$ ones that can be formed (not counting inverses). Some of these may be rare or irrelevant, but others may be important and are indeed used in YAGO (e.g. `bornAtPlace`, `bornOnDate`).
– The choice of the primary pair of entities and their binary relation (John and Mary in Table 1) is arbitrary, and a third party willing to query the KB cannot replicate the choice independently. If their choice is different, they will not obtain any results. A possible solution, which is actually implemented in YAGO2s, is to include the triples for the other pairs and reify them, too, but this adds yet another factor of overhead, besides data redundancy that would complicate updates.
– When two or more different events share the same values for the primary pair of arguments, they will share the same triple, but require separate reifications, producing non-unique triple identifiers. For example, if there are two flight connections between Paris and London with different airlines, the triple `Paris isConnectedTo London` will be reified twice, with two different triple identifiers.

If the triplestore implementation makes use of quads (*http://www.w3.org/TR/n-quads/*), the 4-fold overhead can be avoided (though the underlying storage needs a new column), but the other disadvantages still remain. Quad-based singleton named graphs [15] could be used instead of RDF reification, the problems being the same.

2.3 Subproperties

A recent proposal [22] aims to solve some of the issues with RDF reification by instead declaring a subproperty of the original property in the primary pair, and using this subproperty as the subject for the other arguments of the n-ary relation. This is shown in the *Subproperties* example in Table 1.

While the approach enables us to use RDFS reasoning to obtain the triple with the parent property that relates two of the participants, and also reduces the overhead of RDF reification, it still suffers from the problems mentioned above related to the existence of a primary pair. For one, the non-reified binary relationships for the other pairs cannot be inferred from that subproperty.

2.4 Neo-Davidsonian Representations

Another approach, and the one that we will adapt for FrameBase, is to make use of so-called neo-Davidsonian representations [18, p. 600f.]. This means that we first define an entity that represents the event or situation (also referred to as a *frame*) underlying the n-ary relation. Then, this entity is connected to each of the n arguments by means of a property describing the *semantic role* [13, 23].

Note that the process of converting from the binary representation to the neo-Davidsonian one is also called reification, but this is different from *RDF reification* as discussed earlier. In RDF reification, an entity is defined that stands for a whole triple so that additional triples can be used to describe the reified triple as a unit that represents

a statement. However, in the context of event semantics, reification is used to denote the process by which an entity is defined that refers to the event, process, situation, or more generally, frame, evoked by a property or binary relation. Having done this, additional information about it can then easily be added. Both kinds of reification have in common that a new entity is defined to refer to something that before was not explicitly represented by an entity in the KB, but in one case it is a RDF statement while in the other it is an event.

Advantages. Table 2 compares the neo-Davidsonian approach to the alternatives. These require a lot more triples when several direct binary relations need to be included. In the worst case, $k = \frac{n(n-1)}{2}$ despite discounting reciprocal relations, but even if not all of these relations are relevant, connecting all agents and possibly patients to all other elements would be relevant, which would easily satisfy $k > n$.

Semantic Heterogeneity. Unfortunately, there are different ways of using the neo-Davidsonian approach, with different levels of granularity for the events and the semantic roles, from a very small set of abstract generic ones [28] to more specific ones [4].

The Simple Event Model (SEM) Ontology [32] falls within the category of neo-Davidsonian representation with general roles (see Table 1). It defines four very general entities, *Event*, *Actor*, *Place*, and *Time*. It also establishes a framework for creating more specific ones by extending these, but it does not provide these extensions, nor ways to integrate existing KBs in a way that would solve the problem of semantic heterogeneity. Similarly, LODE (Linking Open Descriptions of Events) [28] specifies only very general concepts such as the four just mentioned.

Freebase [4] is a KB built both from tapping on existing structured sources and via collaborative editing. Although it uses its own formalisms, there are official and third-party translations to RDF. Freebase makes use of so-called *mediators* (also called *compound value types*, CVTs) as a way to merge multiple values into a single value, similar to a `struct` datatype in C. There are around 1,870 composite value types in Freebase (1,036 with more than one instance) and around 14 million composite value instances. While CVTs do not represent frames or events per se, from a structural perspective, they can be regarded as isomorphic to a neo-Davidsonian representation with specific roles (see Table 1). However, Freebase places a number of restrictions on CVTs. For instance, they cannot be nested, and there is no hierarchy or network of them that would for example relate a purchasing event to a getting event.

There is ongoing work to add the modeling of semantic roles to schema.org [1]. Schema.org is an effort sponsored by Google, Yahoo, and Microsoft to establish common standards for semantic markup in Web pages. Currently, the new roles pattern proposal is just a proposed model without a proper role inventory, and schema.org merely targets a small restricted number of domains.

FrameNet [11, 27] is a well-known resource in natural language processing (NLP) that defines over 1,000 *frames* with participants (so-called *frame elements*). For example, the verb *to buy* and the noun *acquisition* are assumed to evoke a commercial transaction frame, with frame elements for the seller, the buyer, the goods, and so on.

Previous work has proposed general patterns for using FrameNet in knowledge representation [12] and converted FrameNet to RDF [24], proposing a way to generate schemas from FrameNet. Similarly, the FRED system [25] for building semantic

representations from natural language can be configured to use FrameNet.

3 System Overview

As we have seen, there are a number of different representations used in KBs. In this paper, we use the linguistic resources FrameNet [11] and WordNet [9] to fully develop an extensive schema for large-scale knowledge representation and integration. The schema is composed of an expressive neo-Davidsonian level that draws on a large common inventory of frames, together with a more concise level of direct binary relations, which is connected to the former by means of inference rules.

3.1 FrameNet-based Representation

The use of FrameNet is motivated by the following considerations.

- FrameNet has a long history and aims at descriptions of arbitrary natural language. It thus provides a relatively large and growing inventory of frames and roles, with a broad coverage of numerous different domains.
- FrameNet comes with a large collection of English sentences annotated with frame and frame element labels. This data led to the task of automatic *semantic role labeling* (SRL) [14] of text, now one of the standard tasks in NLP. This strong connection to natural language facilitates question answering and related tasks.
- While FrameNet's lexicon and annotations cover the English language, its frame inventory is abstract enough to be adopted for languages as different as Spanish and Japanese [29]. This also makes it much more suitable as a basis for knowledge representation than language-specific syntax-oriented SRL resources such as PropBank [19].
- FrameNet provides an reasonable level of granularity for the phenomena that humans care to describe. From a theoretical perspective, there is no universally appropriate single level of reification. Any frame element might be reified on its own, and any two elements of a frame could be connected directly by a predicate. Using FrameNet, we strike a well-motivated balance, at a point that is granular enough to constitute a model for natural language semantics. As we will explain in Sect. 5, we also provide a second level of representation, less expressive but more concise, based on the direct binary predicates between frame elements.

3.2 Overview

For creating the FrameBase schema using FrameNet, we take the following steps, which will be further explained in Sect. 4.

a) **FrameNet–WordNet Mapping.** First, we create a high-precision mapping between FrameNet and another well-known lexical resource called WordNet [9], which will be used to enrich the lexical coverage and relations of the FrameBase schema.
b) **Schema Induction.** We use FrameNet, WordNet, and the mapping to create an RDFS schema for FrameBase that has very wide coverage and is extensible. The schema exploits semantic relations from these components (e.g., synonymy, hyponymy, and perspectivization) to transform the original resources for our lightweight RDFS model.

c) **Automatic Reification–Dereification Mechanism.** We create reification–dereification rules in the form of definite clauses that allow the KB to be queried independently of whether a frame is reified or not, and that may also be used to reduce overhead in the KB.

4 FrameBase Schema Creation

4.1 FrameNet–WordNet Mapping

While FrameNet [11, 27] is the largest high-quality inventory of semantic frame descriptions and their participants, WordNet [9] is the most well-known resource capturing meanings of words in a lexical network, covering for example nouns and named entities missing in FrameNet. WordNet, for instance, serves as the backbone of YAGO's ontology. We propose a novel way of mapping the two resources, which later enables us to integrate both of them into our schema.

WordNet contains synsets, which are sets of sense-disambiguated synonymous words with a given part of speech (POS), such as noun or verb. FrameNet contains lexical units (LUs), which are also POS-annotated words associated to frames. Because of the semantics of the containing frame, lexical units are also disambiguated to a certain extent, though not with the same granularity as in WordNet. Our objective is to map synsets and lexical units with the same meaning, so we can later use this to enrich our FrameNet-based schema with relations and annotations from WordNet.

We choose to map each lexical unit to one and only one synset. While there are some lexical units that could be mapped to more than one synset, this will favor precision, which is desirable for the purpose of obtaining a clean knowledge base. The only cases where this model would be detrimental to precision are those where lexical units do not have any associated synset, but these are few and most can easily be avoided by omitting lexical units with parts of speech not covered in WordNet, such as prepositions.

Our choice allows us to model the mapping as a function $S(l|a, b)$ from lexical units to synsets as in (1). S_l stands for the synsets that have the same lexical label and POS as the lexical unit l, μ_{L} and μ_{G} are the lexical and gloss (definition) overlap, respectively, f yields the corpus frequency of the synset, and a and b are parameters for a linear combination (the third parameter can be omitted because of the argmax function).

$$S(l|a, b) = \underset{s \in S_l}{\operatorname{argmax}}\ \mu_{\mathrm{L}}(l, s) + a \cdot \mu_{\mathrm{G}}(l, s) + b \cdot f(s) \tag{1}$$

The lexical overlap μ_{L} of a lexical unit l and a synset s is the size of the intersection between the POS-annotated words from the lexical units in the same frame as l and the POS-annotated words in s and its neighborhood. We define the neighborhood as the synsets connected by a selection of lexical and semantic pointers such as "See also", "Similar to", "Antonym", "Attribute" and "Derivationally related". This expansion is useful to reduce sparsity and better match the sets with those generated for the lexical units, which due to the different semantics of frames and synsets, may already include these related words.

The gloss overlap μ_{G} is the size of the intersection between the set of words in the definition of the lexical unit and the gloss of the synset. For preprocessing these, we rely on the CoreNLP library [31] to clean XML tags, tokenize, POS-label, and lemmatize the text, and we filter out all words except nouns and verbs.

We trained a and b with a greedy search over several randomized seeds, obtaining optimal values $a = 5, b = 0.13$.

4.2 Schema Induction

We model frames as classes whose instances are the particular events. The frame elements of each frame are properties whose domain is that frame. We create a class hierarchy of frames as follows.

1. **General Frames:** FrameNet's frame inheritance and perspectivization relations are modeled as class subsumption between frames, by means of two specific properties that inherit from `rdfs:subClassOf`, so that both remain distinguishable but contribute to the hierarchy and allow RDFS inference. We additionally declared a top frame for the hierarchy. Inheritance between frame element properties is modeled with a direct subproperty relation.
   Thus, under this model, an instance of the *Commerce_sell* frame with a certain *Commerce_sell-Buyer* x, is also an instance of the *Giving* frame and x is the *Giving-Recipient*, because the first frame inherits from the latter. Likewise, it is also an instance of *Transfer* and x is the *Transfer-Recipient*, because *Giving* is a perspective on *Transfer*.
2. **Leaf Nodes:** Since FrameNet's original frame inventory is coarse-grained and different lexical units like *construction* and *to glue* evoke the same frame, we generate what has occasionally been called a *microframe* model: We transform FrameNet such that every lexical unit is treated as evoking its own separate fine-grained frame, which is made a subclass of the more coarse-grained original FrameNet frame.
3. **Intermediate Nodes** The microframe nodes are very fine-grained, e.g. distinguishing *buy* from *acquire*, while some original frames from FrameNet are very coarse-grained, as mentioned above. For instance, various kinship relationships such as *mother*, *sister-in-law*, etc. are lumped together. This wide range of lexical units may stand in various lexical-semantic relationships without these being indicated, including synonymy, antonymy, or nominalization. The only characteristic they have in common is that, by definition, they evoke a similar kind of situation. Overall, neither the fine-grained nor the coarse-grained levels are ideal for knowledge representation purposes.
   We address this by providing a novel intermediate level composed of *synset-microframes* that group equivalent *LU-microframes* together. For this, we generate a set of directly equivalent synset-microframes for each LU-microframe, and we declare `owl:equivalentClass` predicates between these pairs. This is the only predicate we use that needs inference beyond pure RDFS, but we also include a pair of reciprocal `rdfs:subClassOf`, which is semantically equivalent and leaves the possibility of using any out-of-the-box RDFS inference engine. The clusters are thus defined as the resulting equivalence classes over the set of all microframes.
   These clusters are built in several steps. First, for a given LU, we get the corresponding synsets from the FrameNet–WordNet mapping in Sect. 4.1. In the case of our mapping, the set has no more than one element, but in the general case it could have more. Then, we expand that set by adding all other synsets related by lexical relations reflecting cross-POS morphological transformations: "Derivationally related", "Derived from Adjective", "Participle" and "Pertainym". In general, these lexical relations do not

necessarily imply any close semantics (e.g., *create/make – creature/animal*), but when restricted to synsets all tied to the same FrameNet frame, such cases are normally factored out. The goal of using the lexical relations is linking cross-POS LUs that evoke the same specific situation with a different syntactic form, such as nominalizations (*produce–production*), non-finite verb forms (*produce–produced*), adjectivization, or adverbization.

We also use names, definitions and glosses in FrameNet and WordNet to create text annotations for our schema. We attach lexical forms with `rdfs:label` and definitions and glosses from FrameNet and WordNet with `rdfs:comment`.

5 Automatic Reification–Dereification Mechanism

While frames are convenient for representational purposes, users wishing to query the knowledge base benefit from binary predicates between pairs of frame elements. For example, for a birth event, binary predicates like `bornInPlace` and `bornOnDate` can facilitate querying by offering a more compact and simple representation.

We thus present a novel mechanism to seamlessly convert between frame representations and DBPs. This mechanism can also allow us to avoid materializing frame instances when only two frame elements are needed.

We generate *dereification rules* of the following form:

```
?s BinaryPredicate ?o ← ?f a Frame, ?f FE1 ?s, ?f FE2 ?o
```

Additionally, for each dereification rule there is a converse reification rule so that one can go back from binary predicates to the frame representation. Each direct binary predicate (DBP) has only one set of possible frame and frame elements associated, and therefore chaining reification and dereification rules is an idempotent operation.

We build the reification–dereification rules automatically using the annotations of English sentences given for different LUs in FrameNet, namely the grammatical functions (GFs) and phrase types (PTs) [27] associated with different frame elements in the example sentences of each lexical unit.

For verb-based microframes, FrameNet provides three kinds of GF labels: External Argument (Ext), Object (Obj), and Dependent (Dep). Some of the PT labels that can be found are N, NP, Obj, PPinterrog [27]. We create dereified binary predicates for the pairs of frame elements whose syntactic annotations for some sentence satisfy the creation rules below, using the GF and grammatical PT labels. We list the creation rules below, and add some examples of reification-dereification rules associated to the DBPs created by some of them. The postfixes "-s" and "-o" indicate the data associated to the FEs that fill the first and second arguments of the DBP, or equivalently, the subject and the object of the resulting RDF triple.

- Create DBP with name "CONJUGATETHIRDPERSSING(LU)" if

  (GF-s EQUALS Ext) & (GF-o EQUALS Obj) &

  (PT-o IN { N, NP, Obj, PPinterrog, Sinterrog, QUO, Sfin, Sub, VPing })

  Examples of obtained resulting DBPs and reification-dereification rules:

  `?S :dereif-Forming_relationships-divorces ?O`

$$\leftrightarrow \begin{cases} \texttt{?R a :frame-Forming\_relationships-divorce.v ,} \\ \texttt{?R :fe-Forming\_relationships-Partner\_1 ?S ,} \\ \texttt{?R :fe-Forming\_relationships-Partner\_2 ?O .} \end{cases}$$

```
?S :dereif-Win_prize-wins ?O
```

$$\leftrightarrow \begin{cases} \texttt{?R a :frame-Win\_prize-win.v ,} \\ \texttt{?R :fe-Win\_prize-Competitor ?S ,} \\ \texttt{?R :fe-Win\_prize-Prize ?O .} \end{cases}$$

- Create DBP with name "`is` CONJUGATEPASTPARTICIPLE(LU) `by`" if
  (GF-s EQUALS Obj) & (GF-o EQUALS Subj) &
  (PT-o IN { N, NP, Obj, PPinterrog, Sinterrog, QUO, Sfin, Sub, VPing })

- Create DBP with name "CONJUGATETHIRDPERSSING(LU) PREP" if
  (GF-s EQUALS Ext) & (GF-o EQUALS Dep) & (PT-o EQUALS PP(PREP))

  Examples of obtained resulting DBPs and reification-dereification rules:

```
?S :dereif-Creating-createsFrom ?O
```

$$\leftrightarrow \begin{cases} \texttt{?R a :frame-Creating-create.v ,} \\ \texttt{?R :fe-Creating-Creator ?S ,} \\ \texttt{?R :fe-Creating-Components ?O .} \end{cases}$$

```
?S :dereif-Win_prize-winsAt ?O
```

$$\leftrightarrow \begin{cases} \texttt{?R a :frame-Win\_prize-win.v ,} \\ \texttt{?R :fe-Win\_prize-Competitor ?S ,} \\ \texttt{?R :fe-Win\_prize-Venue ?O .} \end{cases}$$

  For some FEs in this and the next rule, we assign a specific preposition, like "at" for `Time` and "in" for `Place`. For example:

```
?S :dereif-Destroying-destroysAtTime ?O
```

$$\leftrightarrow \begin{cases} \texttt{?R a :frame-Destroying-destroy.v ,} \\ \texttt{?R :fe-Destroying-Cause ?S ,} \\ \texttt{?R :fe-Destroying-Time ?O .} \end{cases}$$

```
?S :dereif-Intentionally_create-establishesInPlace ?O
```

$$\leftrightarrow \begin{cases} \texttt{?R a :frame-Intentionally\_create-establish.v ,} \\ \texttt{?R :fe-Intentionally\_create-Creator ?S ,} \\ \texttt{?R :fe-Intentionally\_create-Place ?O .} \end{cases}$$

- Create DBP with name "`is` CONJUGATEPASTPARTICIPLE(LU) PREP" if
  (GF-s EQUALS Obj) & (GF-o EQUALS Dep) & (PT-o EQUALS PP(PREP))

By using the grammatical subject as subject of the triple, we avoid rules defining certain kinds of DBPs that would be rarely useful, like those connecting the time and place, or the place and the cause.

There is no explicit syntactic annotation in FrameNet to indicate if the example sentences are in passive form. We used two different heuristics for detecting this. One draws on the POS annotations available in FrameNet, and decides that a sentence is in passive iff the target (LU) verb is conjugated as a past participle, and there is a conjugated form of the verb *to be* in a prior position, without another verb in between. The other heuristic uses the Stanford Parser [20]. Both heuristics make type I and II mistakes

differently, so we discarded the cases where they disagree, and for the ones that they agree that they are passive, we created the rules inverting the Ext/Obj GFs.

We restrict ourselves to verb-based microframes, because the process above is more difficult and error-prone with nouns. However, the synset-microframe clustering of our schema already makes many of the morphosemantic variations of a verb, including nominalizations, logically equivalent.

With the rules obtained with the process above, the same DBP can be associated to different pairs of frame elements in a given LU-microframe, owing to different senses or syntactic frames for a given verb (for example the transitive and intransitive frames for *smuggle*). This would conflate different senses, and if the reification and the dereification directions of the rules were chained, it would logically entail different pairs of frame elements, which would not be sound. Furthermore, a given pair of frame elements can also produce different DBPs. To achieve the idempotency mentioned earlier, we use the Kuhn–Munkres algorithm to obtain a one-to-one assignment, using as weights the number of annotated example sentences for a DBP and a pair of frame elements, because the patterns with more example sentences are usually more intuitive. The cubic complexity of the algorithm is not a concern because each frame leads to a separate graph on which we can operate independently.

We have implemented the reification-dereification rules as SPARQL CONSTRUCT queries, due to SPARQL's prominence as a standard query language for KBs. These can be used to materialize the DBPs into the KB. Other options would be possible, such as using a general-purpose inference engine that can handle propositional clauses, like the Rubrik reasoner in Jena [5].

6 Evaluation

We now evaluate the quality of the results and show some example queries.

6.1 FrameNet–WordNet Alignment

To evaluate the created schema, we first compared our FrameNet–WordNet mapping to the MapNet gold standard [30]. MapNet uses older versions of FrameNet and WordNet, so that we had to apply mappings from WordNet 1.6 to 3.0 [7], removing those with a confidence lower than one. For mapping FrameNet 1.3 to 1.5, we removed the few LUs that are not contained in the new version. Table 3 compares the results against state-of-the-art approaches and the scores that they report on the MapNet gold standard. As expected, our approach achieves high precision, while still maintaining good recall. We use 5-fold cross-validation for our results.

6.2 Schema Induction

The FrameBase schema is based on FrameNet and WordNet and our mappings between the two resources. It provides 19,376 frames, including 11,939 LU-microframes and 6,418 synset-microframes, all with lexical labels. A total of 18,357 microframes are clustered into 8,145 logical clusters, which are the sets of microframes whose elements are linked by a logical equivalence relation. The size of the schema is 250,407 triples.

We have obtained an average precision of $87.55\% \pm 6.18\%$ with a 95% Wilson confidence interval. The evaluation showed a small change of nuance for $31.15\% \pm 9.38\%$

	Prec	Rec	F1	Acc
SVM Polyn. kernel 1 [30]	0.761	0.613	0.679	—
SVM Polyn. kernel 2 [30]	0.794	0.569	0.663	—
SSI-Dijkstra [21]	0.78	0.63	0.69	—
SSI-Dijkstra+ [21]	0.76	0.74	0.75	—
Neighborhoods [10]	—	—	—	0.772
Our mapping	0.789	0.709	0.746	0.864

Table 3: Comparison of our FrameNet–WordNet mapping to state-of-the-art approaches in terms of precision, recall, F1, and accuracy

of the correct pairs – most of these are caused by our choice to use semantic pointers such as "Similar to", which could be removed if we desire very fine-grained distinctions of microframes. The precision has been calculated from a random sample of 100 intra-cluster pairs that have been independently annotated by two of the authors. We have obtained the linear weighted Cohen's Kappa over the three-valued combination of the two variables with which we annotate each cluster pair, obtaining a value of 0.23 over a maximum of 0.87. We obtained the scores with a random annotator.

In addition to the number of frames, the FrameBase schema provides a vocabulary of frame elements that goes well beyond the knowledge currently included in most KBs, in particular beyond time and location. This additional knowledge is routinely conveyed in natural language, and we believe that using a schema that provides for it paves the way to include it in KBs, either manually or automatically.

6.3 Reification–Derefication Rules

We also provide 14,930 reification–dereification rules for the same number of direct binary predicates, with both human-readable IRIs and lexical labels. We obtained an average precision of $86.59\% \pm 6.41\%$, and $76.13\% \pm 8.65\%$ of the correct rules were found easily readable. We consider a rule to be not easily readable if the name of the direct binary predicate contains a frame element whose meaning is not obvious for a layman reader, or if it contains a preposition that is appropriate for some but not all possible objects, or it is not appropriate for the frame element in the name. For this evaluation, we followed the same annotation methodology as for the intra-cluster pairs, obtaining a Cohen's kappa of 0.39 over a maximum of 0.54.

6.4 Knowledge Base Integration and Querying

Knowledge from other KBs such as Freebase can be integrated using *integration rules*, which can also be implemented as SPARQL CONSTRUCT queries. The two examples below were created manually.

```
CONSTRUCT {
  _:e a framebase:frame-People_by_jurisdiction-citizen.n .
  _:e framebase:fe-People_by_jurisdiction-Person ?person .
  _:e framebase:fe-People_by_jurisdiction-Jurisdiction ?country .
} WHERE {
  ?person freebase:people.person.nationality ?country . }
```

```
CONSTRUCT {
  _:e a framebase:frame-Leadership-leader.n .
  _:e framebase:fe-Leadership-Leader ?o1 .
  _:e framebase:fe-Leadership-Governed ?o2 .
  _:e framebase:fe-Leadership-Role ?o3 .
  _:e framebase:fe-Leadership-Type ?o4 .
  _:timePeriod a framebase:frame-Timespan-period.n .
  _:timePeriod framebase:fe-Timespan-Start ?o5 .
  _:timePeriod framebase:fe-Timespan-End ?o6 .
} WHERE {
  ?cvti a freebase:organization.leadership .
  OPTIONAL { ?cvti freebase:organization.leadership.person ?o1 .}
  OPTIONAL { ?cvti ...:organization.leadership.organization ?o2 .}
  OPTIONAL { ?cvti freebase:organization.leadership.role ?o3 .}
  OPTIONAL { ?cvti freebase:organization.leadership.title ?o4 .}
  OPTIONAL { ?cvti freebase:organization.leadership.from ?o5 .}
  OPTIONAL { ?cvti freebase:organization.leadership.to ?o6 .} }
```

FrameBase facilitates novel forms of queries. The following query, for instance, uses reified patterns to find the heads of the World Bank. Note that the clusters implemented in RDFS allow searching for the noun *head* (from the leadership frame), although the integration rule above only produced an instance of `fmbs:frame-Leadership-leader.n`. The results in Table 4 show example instances seamlessly integrated into our FrameBase schema from both Freebase (rows 1–3, extracted from the second example integration rule above) and YAGO2s (rows 4–5, extracted with a similar integration rule made for YAGO2s).

```
SELECT DISTINCT ?leader ?role WHERE {
  ?lumfi a fmbs:frame-Leadership-head.n .
  ?lumfi fmbs:fe-Leadership-Governed ?worldBank.
  ?lumfi fmbs:fe-Leadership-Leader ?leader .
  VALUES ?worldBank {yago:World_Bank freebase:m.02vk52z}
  OPTIONAL{ ?lumfi fmbs:fe-Leadership-Role ?role } }
```

Alternatively, a direct binary predicate from the dereification rules can be used to obtain the same non-optional results, as illustrated in the query below. Either *leads* or *heads* can be used because the LU-microframes for these verbs are in the same cluster as the nouns *leader* and *head*, and there is a dereification rule between the *Leader* and *Governed* frame elements for both.

```
SELECT DISTINCT ?leader WHERE {
  ?leader fmbs:dereif-Leadership-heads ?worldBank.
  VALUES ?worldBank {yago:World_Bank freebase:m.02vk52z} }
```

FrameBase can also be applied with natural language processing tools for question

?leader	?role
fb:m/0h_ds2s 'Caroline Anstey'	fb:m/04t64n 'Managing Director'
fb:m/0d_dq5 'Mahmoud Mohieldin'	fb:m/04t64n 'Managing Director'
fb:m/047cdkk 'Sri Mulyani Indrawati'	fb:m/01yc02 'Chief Operating Officer'
yago:Jim_Yong_Kim	--
yago:Robert_Zoellick	--

Table 4: Results from the query

answering and data mining. For example, given the question "Who has been the head of the World_Bank", the SRL tool SEMAFOR [6] successfully extracts the frame *Leadership* with lexical unit *head.noun* and frame elements *Governed* and *Leader*. Based on this, and after a named entity disambiguator like AIDA [17] matches World_Bank to the entities in the KBs, the structured query can easily be built. Moreover, the same procedure can also be used to integrate new knowledge from a text into the KB, like FRED [25] does.

7 Conclusion

FrameBase is a novel approach for connecting knowledge from different heterogeneous sources to decades of work from the NLP community. Events can be described in very different ways across different knowledge bases. Our framework not only provides an efficient model to describe n-ary relations, but also integrates and transforms FrameNet and WordNet to yield a broad-coverage inventory of frames. Additionally, linguistic annotations in FrameNet such as the ones used to create the reification–dereification rules can also be used to generate natural language, for instance, for summarizing a portion of a KB for non-technical users.

Regarding future lines of work, we are currently completing the integration of the instance data from YAGO2s and Freebase into the FrameBase schema, using integration rules such as the examples in Sect. 6.4, but automatically generated. This will lead to the first large-scale FrameNet-based KB. Given FrameBase's close connection to natural language, we also intend to study methods for better adapting semantic role labeling tools to question answering [6]. We are also investigating the ways that FrameBase enables for querying multiple KBs simultaneously with on-the-fly data integration.

Please refer to `http://framebase.org` for information on using FrameBase.

Acknowledgments This research was partially funded by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No. FP7-SEC-2012-312651 (ePOOLICE project). as well as China 973 Program Grants 2011CBA00300, 2011CBA00301, and NSFC Grants 61033001, 61361136003, 20141330245.

References

1. Roles in Schema.org. Technical report, W3C, 2014. http://www.w3.org/wiki/WebSchemas/RolesPattern.
2. C. Bizer, T. Heath, and T. Berners-Lee. Linked data–the story so far. *IJSWIS*, 5(3):1–22, 2009.
3. C. Böhm, G. de Melo, F. Naumann, and G. Weikum. LINDA: Distributed Web-of-data-scale Entity Matching. CIKM '12, pages 2104–2108, 2012.
4. K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. SIGDATA, pages 1247–1250, 2008.
5. J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson. Jena: Implementing the semantic web recommendations. WWW '04, 2004.
6. D. Das, D. Chen, A. F. T. Martins, N. Schneider, and N. A. Smith. Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56, Mar. 2014.
7. J. Daudé, L. Padró, and G. Rigau. Mapping wordnets using structural information. ACL, 2000.

8. L. Del Corro and R. Gemulla. Clausie: Clause-based open information extraction. WWW '13, 2013.
9. C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.
10. O. Ferrández, M. Ellsworth, R. Munoz, and C. F. Baker. Aligning FrameNet and WordNet based on Semantic Neighborhoods. LREC '10, 2010.
11. C. J. Fillmore, C. R. Johnson, and M. R. Petruck. Background to Framenet. *International journal of lexicography*, 16(3):235–250, 2003.
12. A. Gangemi and V. Presutti. Towards a pattern science for the semantic web. *Semantic Web*, 1(1):61–68, 2010.
13. A. Gangemi and V. Presutti. A Multi-dimensional Comparison of Ontology Design Patterns for Representing n-ary Relations. In *SOFSEM '13*, pages 86–105, 2013.
14. D. Gildea and D. Jurafsky. Automatic labeling of semantic roles. *Computational linguistics*, 28(3):245–288, 2002.
15. P. Hayes and P. Patel-Schneider. RDF 1.1 semantics. Technical report, W3C, 2014. http://www.w3.org/TR/rdf11-mt/.
16. J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence* , 194(0):28–61, 2013.
17. J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust Disambiguation of Named Entities in Text. EMNLP '11, pages 782–792, 2011.
18. D. Jurafsky and J. H. Martin. *Speech and Language Processing*. Pearson Prentice Hall, 2nd edition, 2009.
19. P. Kingsbury and M. Palmer. From TreeBank to PropBank. LREC '02, 2002.
20. D. Klein and C. D. Manning. Accurate unlexicalized parsing. ACL '03, pages 423–430, 2003.
21. E. Laparra, G. Rigau, and M. Cuadros. Exploring the integration of WordNet and FrameNet. GWC '10, 2010.
22. V. Nguyen, O. Bodenreider, and A. Sheth. Don't Like RDF Reification?: Making Statements About Statements Using Singleton Property. WWW '14, 2014.
23. N. Noy and A. Rector. Defining N-ary Relations on the Semantic Web. W3C Working Group Note, W3C Consortium, April 2006. http://www.w3.org/TR/swbp-n-aryRelations/.
24. A. G. Nuzzolese, A. Gangemi, and V. Presutti. Gathering lexical linked data and knowledge patterns from FrameNet. K-CAP '11, pages 41–48, 2011.
25. V. Presutti, F. Draicchio, and A. Gangemi. Knowledge Extraction Based on Discourse Representation Theory and Linguistic Frames. In *EKAW'12*, pages 114–129. 2012.
26. J. Rouces. Enhancing Recall in Semantic Querying. volume 257 of *SCAI '13*, page 291, 2013.
27. J. Ruppenhofer, M. Ellsworth, M. R. Petruck, C. R. Johnson, and J. Scheffczyk. *FrameNet II: Extended Theory and Practice*. ICSI, 2006.
28. R. Shaw, R. Troncy, and L. Hardman. LODE: Linking Open Descriptions of Events. In *ASWC '09*, Lecture Notes in Computer Science, pages 153–167, 2009.
29. C. Subirats. Spanish FrameNet: A frame-semantic analysis of the Spanish lexicon. In *Multilingual FrameNets in Computational Lexicography: Methods and Applications*. Mouton de Gruyter, 2009.
30. S. Tonelli and D. Pighin. New Features for FrameNet: WordNet Mapping. CoNLL '09, pages 219–227, 2009.
31. K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. HTL-NAACL '03, 2003.
32. W. R. Van Hage, V. Malaisé, R. Segers, L. Hollink, and G. Schreiber. Design and use of the Simple Event Model (SEM). *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(2):128–136, 2011.
33. M. Yahya, K. Berberich, S. Elbassuoni, M. Ramanath, V. Tresp, and G. Weikum. Natural language questions for the web of data. EMNLP-CoNLL '12, 2012.