Process Extraction from Texts using Semantic Unification

Konstantin Sokolov[1,2], Dimitri Timofeev[3] and Alexander Samochadin[3]
[1]*Mobile Device Management Systems Lab, Peter the Great St. Petersburg Polytechnic University, St. Petersburg, Russia*
[2]*Department of Mathematical Linguistics, St. Petersburg State University, St. Petersburg, Russia*
[3]*Institute of Computer Science and Technology, Peter the Great St. Petersburg Polytechnic University, St. Petersburg, Russia*
vtqveant@gmail.com, dtim@dcn.icc.spbstu.ru, samochadin@gmail.com

Keywords: Business Process Management, Process Extraction, Text Mining, Natural Language Processing

Abstract: In order to extract a process model from natural language sources, process elements such as agents, resources or actions have to be identified across one or multiple texts. Natural language descriptions tend to provide only partial and potentially contradictory information. Inability to systematically reconcile partial information significantly limits capabilities of current systems. We propose to address this problem by means of semantic unification integrated in the common text processing pipeline.

1 INTRODUCTION

Providing a formal specification of a business process requires substantial work and high qualification from the analyst. While the role of human experts in analysis and modeling of business processes is crucial, their activity may be substantially automated. It is especially important when business process modeling is performed in the enterprise environment. Large organizations have lots of processes but, typically, the processes are already specified textually in the form of instructions, user guides, manuals, and standards. Using text documents as a source of process models lowers the cost of business process management. It can also give new insights into the existing processes by introducing new sources of knowledge, such as customers claims, reviews, user stories, or other user-generated content on corporate websites and in social media.

Our interest in applying process extraction techniques to natural language texts is motivated by the task of development of process-oriented mobile services. This class of services includes:

- specialized step-by-step guides that inform users on actions that should be performed to achieve a goal (e.g., to obtain a visa, or to apply to the university);
- enterprise-level business process supporting software;
- mobile assistants that use procedural knowledge to answer questions and schedule events.

This research is part of a project aiming to design and implement a service platform based on a mobile device management system. The platform provides a set of process-oriented services, which require development of process models. Automated process extraction can greatly simplify service deployment, especially in organizations like universities, where most processes are specified in normative documents, but formal process models are rarely available. Besides that, users of process-oriented mobile services may benefit from the ability to share their knowledge of processes and to use these community contributions.

In this position paper we discuss the limitations of existing text processing pipelines used for process extraction, especially with respect to handling multiple documents, and propose an approach to systematically overcome these limitations. The rest of the paper is structured as follows. In Section 2 existing methods of process extraction from text are reviewed. In Section 3 we discuss the limitations of current approaches and propose a way to overcome them by using semantic unification. Section 4 describes the core ideas behind our method. Section 5 discusses how semantic unification fits into the text processing pipeline, and provides an example. The section 6 concludes.

2 RELATED WORK

A method of building a process model from a set of unstructured documents should provide solutions to the following problems: how to decide what entities

should go into the model, how superficially unrelated data dispersed in the texts should be combined, and how to obtain a formal model of a process.

The CREWS model (Achour, 1998) is often used to formulate a conceptual description of a process. The model specifies two kinds of building blocks, objects (agents or resources) and actions (atomic actions or action flows). In a text objects are generally represented by noun phrases, and actions by verb phrases. Extraction of entities and relations is a standard topic of natural language processing, therefore it is rather straightforward to integrate both technologies in order to match a process with a text that describes it. In the R-BPD framework prototype (Ghose et al., 2007), constituents of a process model are extracted with a combination of template matching, part-of-speech tagging, and phrase chunking. Other systems take advantage of more sophisticated natural language processing techniques. The use case analysis engine by Sinha et al. (2008) implements a pipeline that involves lexical processing, parsing, dictionary-based concepts annotation, anaphora resolution, context annotation, and building of process models. A similar method is implemented by Gonçalves et al. (2009, 2011). Both rely on problem-oriented grammars for shallow parsing. Friedrich et al. (2011) use a wide-coverage Stanford parser (Manning, 2003), and utilize WordNet (Miller, 1995) and FrameNet (Baker et al., 1998) databases to obtain information about semantic relations, including synonymy. Ackermann and Volz (2013) describe a prototype system based on recursively defined templates which is able to extract domain models from text. They adopt feature structures to model both the templates and the results of natural language analysis, and apply a variant of unification algorithm to perform the matching between the two. This approach allows to map text fragments to elements of an intended domain model and to interactively modify templates based on user feedback. In general, their system is pattern-based and uses unification techniques for comparison of tree structures of syntactic nature.

3 LIMITATIONS

It should be obvious from the above, that the ability of such systems to cope with intricacies of natural language is limited by the tools that are used to produce representations of entities and events that comprise a process. Common approaches to extraction of a process model from a document rely on a typical text processing pipeline, which includes text normalization, segmentation, tokenization, morphological analysis, named entity recognition, and syntactic parsing. Some systems additionally perform semantic parsing to produce trees labeled with semantic roles of the constituents. In order to handle pronouns an anaphora resolution step may be added to the pipeline. All systems process each sentence independently.

A serious limitation of current approaches is their inability to consistently handle multiple descriptions of the same process. Process descriptions supplied by users contain errors due to the lack of proof-reading and tend to describe only those parts of processes that their authors are aware of or interested in. Documents by different authors can be inconsistent with respect to their vocabulary. As a result, these systems are generally designed to extract a model of a single process from a single document. Currently, processing multiple sources requires generation of models for each of the documents with a subsequent merging by a human expert, which should be automated. The common vocabulary is even more important in an enterprise environment, where other knowledge-based systems are available. The process extraction system should be able to align extracted entities with existing ontologies.

To create a formal model of a process, a set of hand-crafted rules is used by each system. This approach is perfectly valid for building models with a known structure. However, the facts that do not fit into the structure are ignored, making such systems less convenient for exploratory analysis, and restricting the number of texts that can be processed. Further, a known limitation of systems that use explicit rules is the difficulty of rule modification. It is desirable to have simpler rules that could be learned from corpora, and to be able to combine them with higher-level process analysis algorithms.

In sum, to overcome the limitations of the current approaches it should be made possible to automatically construct entity and event representations that could easily integrate in the existing natural language processing pipelines, support easy alignment with external knowledge bases and ontologies, be expressive enough to allow for flexible data integration across the limits of a sentence, and be formally compatible with existing frameworks of process extraction. To achieve this we propose a method of semantic unification for entity resolution and data reconciliation based on feature structures. To build a process model, we adopt the methods of process mining (van der Aalst, 2011), where the text is translated into a sequence of ordered events, and a process mining algorithm is invoked to infer a model that fits this sequence.

4 SEMANTIC UNIFICATION

4.1 Feature Structures

The theoretical background of using feature structures for semantic representation is the situation semantics of Barwise and Perry (1983). This approach requires that some extralinguistic knowledge be included in the context of an utterance for it to be coherent and non-contradictory. This follows from the fact that interpretation is impossible without relating to contextual information, like time and place of an utterance, propositional attitude of the speaker, etc. Instead of linguistic production it is rather the context of observation of linguistic performance that has to be taken into account during interpretation. This approach differs from the traditional model-theoretic view of formal semantics in that the meaning is not related to the whole information about the world, but instead it is understood as a structured representation, which only partially describes the world and consequently is provided with a partial interpretation. The situation is thus only a fragment of the model structure, which should be merged with other such fragments in the presence of extralinguistic data. Following Pollard and Sag (1987), the formal description of situations is given in the form of feature structures, for which an operation of unification is defined, where unification serves as a means of obtaining a full description from partial ones by combining linguistic information and extralinguistic knowledge as long as it is possible to do so without falling into contradiction.

Feature structures is a well-known formalism which has been applied to knowledge representation and formulation of various grammatical theories based on unification, including Head-driven Phrase Structure Grammar (Pollard and Sag, 1994) and Lexical Functional Grammar (Kaplan and Bresnan, 1982). Feature structures are recursively defined and can support internal references. They can be represented in multiple forms, the traditional attribute-value matrix (AVM) notation of computational linguistics and the direct rendering of an underlying graph structure, which is a natural representation for conceptual graphs, RDF, etc. A number of extensions is commonly used, including support for disjunctive features and typing constraints. Standard definitions of typed feature structures (TFS) can be found in (Copestake, 2000), and the more extended treatment of formal aspects of the underlying theory in (Carpenter, 2005) and (Francez and Wintner, 2011). Here we limit ourselves to a few examples.

The following AVM represents an object of type (or sort) *noun*, which has two attributes, a *number* and an *orthography*.

$$\begin{bmatrix} \textit{noun} & \\ \text{NUMBER} & \text{sg} \\ \text{ORTHOGRAPHY} & \text{foot} \end{bmatrix}$$

The other AVM represents another possible description which differs from the first one in that it has a different set of attributes.

$$\begin{bmatrix} \textit{noun} & \\ \text{CASE} & \text{nom} \\ \text{ORTHOGRAPHY} & \text{foot} \end{bmatrix}$$

Note, however, that the values of common attributes are the same, so the information contained in both descriptions can be combined (or unified) to result in a single description.

$$\begin{bmatrix} \textit{noun} & \\ \text{CASE} & \text{nom} \\ \text{NUMBER} & \text{sg} \\ \text{ORTHOGRAPHY} & \text{foot} \end{bmatrix}$$

The unification process is restricted by the types that are assigned to particular feature structures. The types form a pre-order, so that a more specific type can license combination of a number of more general partial descriptions of subsuming types. Thus a set of typed feature structures resembles a system of objects provided with a mechanism of multiple inheritance.

Feature structures can be regarded as abstract entities and be given a formal logical description, which is similar to the knowledge representation approach of Semantic Web. The earliest proposals along these lines are the Kasper-Rounds logic (Kasper and Rounds, 1986), which is close to the constraint-based system PATR-II, and an approach of Gazdar et al. (1988). Johnson (1991a,b) uses a decidable fragment of first order logic (Bernays–Schönfinkel class), whereas Blackburn (1993) builds a formalism based on hybrid logic, an extension of modal logic. The latter approach was used by Baldridge and Kruijff (2002) for Hybrid Logic Dependency Semantics (HLDS), a semantic representation formalism used in the OpenCCG[1] system. A connection with description logics is also well established. Blackburn (1993) notices that both feature logics and description logics are in essence modal logics with superficial differences of the way they are formulated. What is common to all these approaches is the property of decidability, that the authors of corresponding logics strive to maintain.

[1] `http://openccg.sf.net`

4.2 Semantic unification

Feature structures can serve as a basic formalism in the entity resolution task for the following reasons.

The type pre-order of TFS provides the means of relating feature-based descriptions to an ontology. Krieger and Schäfer (2010) describe a method of translating an OWL ontology into TFS and vice versa. There an additional knowledge is used to improve recall and precision of named entity recognition algorithms. In general, domain knowledge can be used in the process of text analysis, whereas the results of entity resolution can be exported to an ontology.

Feature structures allow for incremental construction of a full description of an object based on its partial descriptions. Semantic representations based on feature structures have been used in many approaches in computational linguistics, including Minimal Recursion Semantics (Copestake et al., 2005), formal representations of lexical semantics in Generative Lexicon (Pustejovsky, 1991), Hybrid Logic Dependency Semantics (Baldridge and Kruijff, 2002) and others. In these approaches the process of unification is used to model semantic composition.

A serious limitation of the symbolic unification approach of the aforementioned proposals is its inability to account for the variations of values present in the corresponding semantic representations. Minor modifications of spelling may result in failure of determining whether multiple realizations of the same attribute correspond to the same value and thus block the unification process. Analogously, in cases of inter- and intrasentential anaphora pronouns should be linked to the expressions they refer to in other segments of text, which is problematic to do at the purely symbolic level. Recently Abramsky and Sadrzadeh (2014) adopted an approach similar to ours for anaphora resolution using discourse representation structures as their semantic formalism. An important aspect of their work is that instead of maintaining global coherence of semantic structures they propose to consider semantic coherence locally, modeling it with sheaf theory, where data integration is regarded as gluing of semantic representations. This process generally results in multiple variants of data combination, each of which can be assigned a probability based on its realization in a corpus. By introducing an empirically sound quantitative measure of appropriateness of particular way of combining partial semantic information it is possible to alleviate the problems of purely symbolic unification.

An apparent discrepancy between the traditional feature structure unification and the semantic unification approach based on sheaf theory can be made clear by analogy with the general unification theory (Baader and Snyder, 2001), where a distinction is made between purely syntactic and equational unification. Whereas the former relies on unconditioned term substitution, the latter requires that the possibility of term substitution be controlled by certain external conditions, thus resulting in unification modulo an equational theory. The semantic unification process as applied to entity resolution can be viewed as an extension of the traditional symbolic unification of feature structures augmented with a form of empirically motivated corpus-based equational theory.

5 PROCESS EXTRACTION

To build a description of a business process, we extract from each text a set of relations that represent actions and events. In the simplest case these relations have the form $\langle$NP, VP$\rangle$, where the noun phrase NP correspond to the agent, and the verb phrase VP describes the action. More complex relations include other semantic roles such as theme or goal. For example, an argument structure for the sentence "The seller gives the invoice to the customer" can be described as follows.

$$\begin{bmatrix} \text{ACTION} & \text{give} \\ \text{AGENT} & \text{seller} \\ \text{THEME} & \text{invoice} \\ \text{RECEPIENT} & \text{customer} \end{bmatrix}$$

Technically, attribute values are not plain text segments, but rather references to feature structures attached to corresponding segments of parsed text. These structures may contain both semantic and syntactic attributes.

The model is inferred from a log, a sequence of actions grouped by case. Process logs are built from the relations extracted during the previous step. To do so, we take all relations that describe actions, and sort them by time under the heuristic that the corresponding actions are performed in the order of their mention in the text, which is a plausible assumption for descriptions of processes. Explicit time attributes may change this order according to specified rules. We use these attributes to generate a set of cases. If all actions are mandatory, and there are no alternatives, a single linear case is generated. Each optional action doubles the quantity of the cases, so that one half of the resulting cases does not contain the action as it never occurs. Alternatives are handled similarly. The model may be inferred from the resulting set of cases using process mining algorithms (currently we

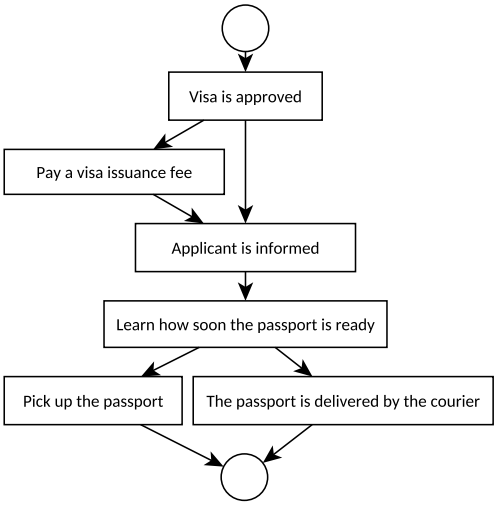

Figure 1: A graph of the inferred process.

use the ProM tool[2]).

As an example, consider a paragraph from a guide for obtaining the United States visa.

"*When the visa is approved, you may pay a visa issuance fee if applicable to your nationality, and will be informed how your passport with visa will be returned to you. Review the visa processing time, to learn how soon your passport with visa will generally be ready for pick-up or delivery by the courier.*"

Suppose the following actions have been extracted.

1. *Visa is approved* (mandatory).
2. *Pay a visa issuance fee* (optional).
3. *Applicant is informed* (mandatory).
4. *Learn how soon the passport is ready* (mandatory).
5. *Pick up the passport* (alternative).
6. *Passport is delivered by the courier* (alternative).

There are one optional action and one action with two alternatives, so we get four process cases: (1, 2, 3, 4, 5), (1, 2, 3, 4, 6), (1, 3, 4, 5), and (1, 3, 4, 6).

The inferred process is presented in Figure 1.

6 CONCLUSION

In this paper we described an approach of extraction of process models from texts. Currently we consider English texts only. The method itself is not limited to one language, but linguistic resources, especially grammars and lexical databases, are required to support other languages. Our main goal is to handle multiple documents with partial descriptions of a process. We propose to add to the processing pipeline an entity resolution step implemented as the semantic unification of typed feature structures. This approach also allows for integration of text processing algorithms with domain knowledge represented as ontologies.

The work presented is in the early stage of development, and we are currently working on a prototype and on the evaluation of our method.

[2] http://www.promtools.org

ACKNOWLEDGEMENTS

This research is a part of the joint project by IBS (Moscow, Russia) and Peter the Great St. Petersburg Polytechnic University (St. Petersburg, Russia). This work is financially supported by the Ministry of Education and Science of the Russian Federation (state contract 02.G25.31.0024 from 12.02.2013).

REFERENCES

Abramsky, S. and Sadrzadeh, M. (2014). Semantic unification. In *Categories and Types in Logic, Language, and Physics*, pages 1–13. Springer.

Achour, C. B. (1998). Guiding scenario authoring. In *European-Japanese Conference on Information Modelling and Knowledge Bases*, page 1.

Ackermann, L. and Volz, B. (2013). model [nl] generation: natural language model extraction. In *Proceedings of the 2013 ACM workshop on Domain-specific modeling*, pages 45–50. ACM.

Baader, F. and Snyder, W. (2001). Unification theory. *Handbook of automated reasoning*, 1:445–532.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley FrameNet project. In *COLING-ACL '98: Proceedings of the Conference*, pages 86–90, Montreal, Canada.

Baldridge, J. and Kruijff, G.-J. M. (2002). Coupling CCG and Hybrid Logic Dependency Semantics. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, July 2002*, pages 319–326.

Barwise, J. and Perry, J. (1983). Situations and attitudes.

Blackburn, P. (1993). *Modal logic and attribute value structures*. Springer.

Carpenter, B. (2005). *The logic of typed feature structures: with applications to unification grammars, logic programs and constraint resolution*, volume 32. Cambridge University Press.

Copestake, A. (2000). Appendix: Definitions of typed feature structures. *Natural Language Engineering*, 6(01):109–112.

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.

Francez, N. and Wintner, S. (2011). *Unification grammars*. Cambridge University Press.

Friedrich, F., Mendling, J., and Puhlmann, F. (2011). Process model generation from natural language text. In *Advanced Information Systems Engineering*, pages 482–496. Springer.

Gazdar, G., Pullum, G. K., Carpenter, R., Klein, E., Hukari, T. E., and Levine, R. D. (1988). Category structures. *Computational Linguistics*, 14(1):1–19.

Ghose, A., Koliadis, G., and Chueng, A. (2007). Process discovery from model and text artefacts. In *Services, 2007 IEEE Congress on*, pages 167–174. IEEE.

Gonçalves, J. C. A. R., Santoro, F. M., and Baião, F. A. (2009). Business process mining from group stories. In *Proceedings of the 2009 13th International Conference on Computer Supported Cooperative Work in Design*.

Gonçalves, J. C. A. R., Santoro, F. M., and Baião, F. A. (2011). Let me tell you a story — On how to build process models. *Journal of Universal Computer Science*, 17(2):276–295.

Johnson, M. (1991a). Features and formulae. *Computational Linguistics*, 17(2):131–151.

Johnson, M. (1991b). Logic and feature structures. In *IJCAI*, pages 992–996.

Kaplan, R. M. and Bresnan, J. (1982). Lexical-functional grammar: A formal system for grammatical representation. *Formal Issues in Lexical-Functional Grammar*, pages 29–130.

Kasper, R. T. and Rounds, W. C. (1986). A logical semantics for feature structures. In *Proceedings of the 24th annual meeting on Association for Computational Linguistics*, pages 257–266. Association for Computational Linguistics.

Krieger, H.-U. and Schäfer, U. (2010). DL meet FL: a bidirectional mapping between ontologies and linguistic knowledge. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 588–596. Association for Computational Linguistics.

Manning, D. K. C. D. (2003). Natural language parsing. In *Advances in Neural Information Processing Systems 15: Proceedings of the 2002 Conference*, volume 15, page 3. MIT Press.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Pollard, C. and Sag, I. (1987). Information-based syntax and semantics, vol. 1.

Pollard, C. and Sag, I. A. (1994). *Head-driven phrase structure grammar*. University of Chicago Press.

Pustejovsky, J. (1991). The generative lexicon. *Computational linguistics*, 17(4):409–441.

Sinha, A., Paradkar, A., Kumanan, P., and Boguraev, B. (2008). An analysis engine for dependable elicitation on natural language use case description and its application to industrial use cases. *IBM Research Report RC24712 (W0812-106)*.

van der Aalst, W. M. P. (2011). *Process Mining, Discovery, Conformance and Enhancement of Business Processes*. Springer-Verlag, Berling Heidelberg.