Anaphora resolution in a pipes-and-filters framework for workflow extraction

Pol Schumacher, Mirjam Minor, and Eric Schulte-Zurhausen

Goethe Universität Frankfurt - Institut für Informatik
D-60325 Frankfurt am Main, Germany
[schumacher|minor|eschulte]@cs.uni-frankfurt.de

Abstract. A central issue in Textual Case-Based Reasoning (TCBR) is the creation of structured case representations from text. Usually these systems apply a Bag-of-Words indexing approach, only few use more advanced methods. We aim at combining TCBR and the recently emerged process oriented Case-Based Reasoning (POCBR). Therefore we developed a workflow extraction framework which allows deriving a formal representation based on workflows (wf) from textual process description. The framework is based on a pipes-and-filters architecture and uses NLP tools to perform information extraction steps. Besides these standard tasks, our framework is able to incrementally create wf models by analysing and modifying the wf models. In detail we present the step of anaphora resolution. Anaphora resolution is a part of the creation of the data-flow, which shows the flow of the data-objects through the wfs. We performed an evaluation of the data-flow for 37 workflows.

1 Introduction

The creation of structured case representations from text has been a central research issue in Textual Case-Based Reasoning (TCBR) for several years [1]. Many Bag-of-Words indexing approaches have been developed based on information extraction methods [2, 3]. A couple of TCBR approaches extract information beyond Bag-of-Words in order to facilitate adaptation of text [4] and more advanced similarity measures for case retrieval[5]. Recently, the extraction of workflows (wfs) for structured case representations has been introduced in research on process-oriented Case-Based Reasoning (POCBR) systems [6, 7]. Traditionally, *workflows* are "the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules" [8]. In a broader notion, wfs describe any flow of activities not necessarily in the sense of a classical business process that involves several participants. This more general view on wfs is used during the paper. A wf consists of a set of *activities* combined with *control-flow structures* like sequences, parallel or alternative branches, and loops. In addition, activities consume *resources* and create certain *products*, which both can be physical matter (such as cooking ingredients or a physical business card) or information. Samples for wfs are the step-by-step creation of a

business card, the processing of an order from a customer or a cooking procedure for preparing a meal.

The extraction of wfs from textual descriptions requires Natural Language Processing (NLP) [9]. NLP tool-chains apply a set of transformations to a stream of text data called *filters*. However, standard NLP tool-chains [10, 11] are not adequate for wf extraction since they do not cover wf-specific filters. Further, the adaptation of a standard NLP tool-chain to a new text corpus is quite laborious. Wf extraction requires a certain degree of flexibility. As an example, the source for the extraction might be a Web page with semi-structured content like cooking recipes. If a new Web page with a slightly different structure of the content is integrated, some of the extraction filters should be exchanged or omitted. We identified a research gap addressing a flexible, partly domain-specific tool-chain dedicated to wf extraction. This paper presents on a novel filter-and-pipe framework for the extraction of wfs from textual process descriptions. This paper is organized as follows. In the next section we describe the framework and a short description of the application in the cooking domain. The subsequent section introduces the data-flow creation and anaphora resolution (see section 3) approaches followed by an evaluation and the discussion of the results. The paper ends with the related work, a short conclusion and an outlook on our future work.

2 Workflow extraction framework

Systems which process natural language need to be flexible and extensible. While there are a lot of systems for generic NLP tasks, there is none for wfs. Therefore we developed a framework which should support the flexibility that is needed for such an application. The framework is based on a pipes-and-filters architecture.

2.1 Workflow representation

The target of wf extraction is a formal representation of the wf in a wf description language. Our wf description language is block oriented and based on the data-model of the CAKE framework[1], this enables us to use the built-in functionality of the framework for Case-Based Reasoning [12]. A wf consists of the control-flow and the data-flow. The control-flow describes the order in which activities are executed. An activity processes resources like information or ingredients. The simplest form of a control-flow is a sequence of activities. A sequence can contain an XOR-, AND-, or LOOP-blocks. These building blocks cannot be interleaved but they can be nested. In addition an activity has a set of semantic descriptors, resources and products. A semantic descriptor is for example the name of the task or additional information which describes "how" a task should be performed, e.g. "for 10 minutes". Resources contain a set of semantic information, these describe additional information about the resources, e.g. amounts or if a resource should be preprocessed like "chopped".

[1] Collaborative Agile Knowledge Engine

2.2 Information extraction software

The framework uses the information extraction software SUNDANCE (Sentence UNDerstanding ANd Concept Extraction) developed by Ellen Riloff [13]. SUNDANCE performs the usual NLP task like tokenization or part of speech tagging but we use SUNDANCE because it has good balance between coverage and robustness.

The SUNDANCE parser assigns syntactic roles (subject, direct object, and indirect object) to text snippets based on a heuristic. Then the SUNDANCE information extraction engine tries to fill a case frame as follows. Each case frame specifies a trigger phrase. If the trigger phrase is detected in a sentence the according case frame is activated. This means that the activation functions of the frame try to match the specified linguistic patterns with the syntactic roles. A slot specifies a syntactic role whose content is extracted from the text.

2.3 Extraction pipeline

The framework is based on a pipes-and-filters [14] architecture. Such an application is a sequence of filters which are connected by pipes. A filter is a self-contained element which performs a data transformation step on the data-stream. The pipes channel the data-stream from the output of a filter to the input of the subsequent filter. A data-stream is sent through this pipeline and each filter is applied to the stream.

At the beginning of the pipeline, the case initially consists of the textual process description. While the case passes through the pipeline it is enriched with additional structure. At the end of the pipeline we have a complete case consisting of the textual process description and the formal wf representation.

Our framework extends the original pipes-and-filters architecture. We allow two different types of filters. The first one, the so called *local filters* operate with a focus on one case. The second one, the *window filters* collect a part of the case-stream (e.g. 5000 cases) and operate on that. The model of a window filter is necessary, because the framework processes a stream of cases which is potentially infinite. The intention is to employ statistical methods for a larger number of textual process descriptions. The statistical approaches benefit from the pipes and filter principle because we employ them on processed data. This intermediate data is the result of the preceding steps of the extraction pipeline. It contains less noise and has more structure than the raw input data.

Figure 1 shows a sample pipeline for the cooking domain. The different filters are created manually the details are described in [6].

3 Data-flow creation and anaphora resolution

In this section we introduce our method which is used to resolve anaphoras in a cooking wf. An anaphora is a linguistic entity which indicates a referential tie to some other entity in the same text [15]. The anaphora resolution is necessary

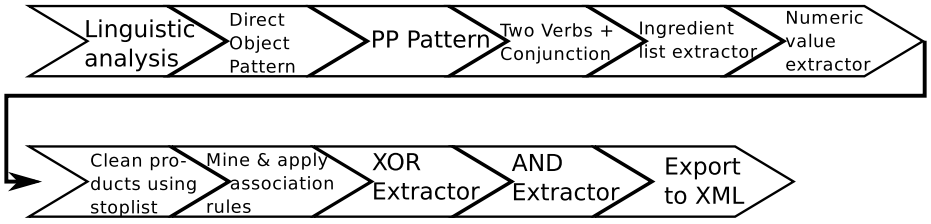

Fig. 1: Overview of the extraction pipeline for the cooking domain.

to complete the data-flow. Several approaches exist to perform the anaphora resolution. Our anaphora resolution approach is based on frequent sequential pattern. We chose this approach because it does not need a complex ontology. We are going to describe the mining and the application of those patterns.

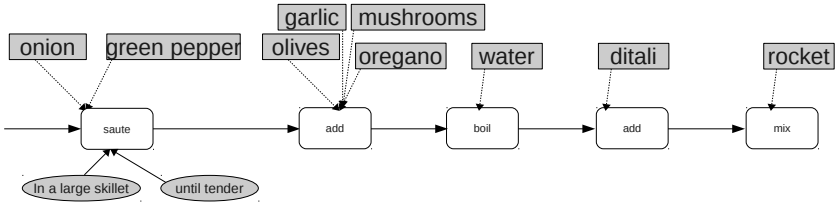

Fig. 2: Sample workflow for cooking rocket pasta with vegetable.

Listing 1.1: Sample transactions for workflow in Fig. 2.

```
WorkflowId,TransactionTime,{Items}
0,0,{Ditali,rocket,water,herbs,
     onion,mushrooms,garlic,
     oregano,green pepper}
0,1,{onion,green pepper}
0,2,{garlic,mushrooms,olives,
     oregano}
0,3,{water}
0,4,{ditali}
0,5,{rocket}
```

Listing 1.2: Top 10 pattern.

```
0.025:<item=butter><item=dough>
0.024:<item=butter><item=mixture>
0.024:<item=flour><item=batter>
0.021:<item=eggs><item=batter>
0.021:<item=flour><item=dough>
0.018:<item=yeast><item=dough>
0.016:<item=baking powder><item=batter>
0.016:<item=butter><item=batter>
0.015:<item=garlic><item=mixture>
0.015:<item=vanilla><item=batter>
```

3.1 Mining sequential pattern

We use a method that was presented by Agrawal [16] to mine sequential patterns in a sequence of transactions. Listing 1.1 displays the transactions which are created from the wf in Fig. 2. It assumes that the wf has the id *0*. The items at the transaction at time *0* are taken from the corresponding ingredient list of

the original recipe. For the details of the sequential pattern mining algorithm we refer to the original paper [16]. The algorithm delivers a set of sequential patterns (see Listing 1.2) with a corresponding support value. The minimum support value which we use is *0.005*. This value is domain dependent and needs to be tuned for a specific domain.

3.2 Creation of data-flow

A sequential pattern can be seen as an association rule. The left side of the rules is the first item-set and the right side of the rule the second item-set of a sequential pattern. A sequential pattern like e.g. $\langle(groundbeef, tomato), (pastasauce)\rangle$ would result in an association rule $\{ground\ beef, tomatoes\} \Rightarrow \{pasta\ sauce\}$. Two observations about anaphoras in cooking wf can be made. Firstly anaphoras are not enumerated in the resource list. Secondly resources of an anaphora are used before the anaphora or are included in the resource list.

The first observation enables us to delete a lot of unnecessary rules. We can delete all rules whose right side contains an ingredient.

The creation of the data-flow is a two phase procedure. The first phase is related to the extraction of tasks. Activities are usually extracted with a set of resources related to them. These resources are used as products. The second phase is the anaphora resolution and the creation of products. We implemented three different approaches which we are going to introduce.

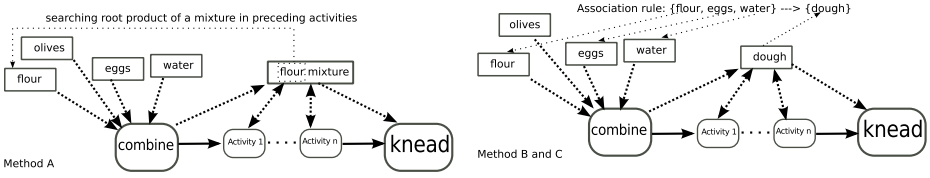

Fig. 3: Illustration of M. A and M. B.

Method A(M. A, see Fig. 3) is based on the observation, that a lot of anaphoras contain the token *"mixture"*. This approach searches for resources which contain this token. If such a resource is found, we copy the resource and delete the token *mixture*. Now the root resource is produced. In the next step we scan the resources of the preceding activities for this root resource. For those activities we check if they contain multiple resources. In that case we create a sole product *root-product + "mixture"* for that activity. At the end we complete the data-flow in the way that products are copied as resources to the next activity. For Fig. 3 the resource *"flour mixture"* is found at the activity *"knead"*. After deleting the token *"mixture"* we get the resource *"flour"*. We search for the resource flour in the preceding activities and we find it at the activity *"combine"*

and the activity uses four resources. We assume now, that the activity combine produces the product *"flour mixture"*

Method B (M. B, see Fig. 3) includes M. A. In addition it iterates over all activities, starting at the first. For each resource of the activity, it is checked if there is an association rule with a matching right side. If such a rule is found, it is looked if one of the previous activities has a resource matching the left side of the rule. We assume then that the anaphora enters the wf at the activity where the left side of the rule is found. Therefore we add the right side of the rule(the anaphora) as sole product of that activity. In the case that multiple rules are found, the one with the best support value is chosen. In Fig. 3 the search for a right side of a rule results with the resource *"dough"*. After a match, the algorithm searches the preceding activities for the resources of the left side of the rule *"flour, eggs, water"*. These resources are linked to the activity *"combine"*. Although the activity has more resources than the left side of the rules enumerates (the resource *"olives"* is not included in the rule) the sole product *"dough"* is created for the activity *"combine"*.

Method C (M. C)includes M. B. A domain specific list of items is added. This list in order to filter out items which were incorrectly extracted as resource. In the cooking domain this list is used to filter out cooking ware.

4 Evaluation

This section describes our hypotheses and our data-flow evaluation approach. The performance of the different methods was measured using the standard evaluation functions Precision, Recall and F-measure [17]. We tested three hypotheses. The first one is that the data-flow created by M. A has the best precision in comparison to M. B and M. C. The second one states that the data-flow created by M. B has a better recall than M. A. The last one declares that the data-flow created by M. C has the highest F-measure in comparison to M. B and M. C.

4.1 Experimental Setup

The experiment was performed on a set of 37 recipes. These recipes were selected randomly from a set of 36 898 recipes which were crawled from a cooking community website[2]. A human expert modelled the data-flow for the recipes in the test set. This served as the golden standard for the evaluation. As the evaluation aimed at the data-flow our expert got the control-flow which was automatically extracted as framework for the golden standard wf. This approach eliminated the paraphrasing and granularity problem of the control-flow. The expert was allowed to use all resources and products that she thought should be in the data-flow, even if they were not mentioned in the text. We got a semantically correct data-flow. The only constraint was, that the expert was not allowed to

[2] www.allrecipes.com

Recipe	P_A	P_B	P_C	R_A	R_B	R_C	F_A	F_B	F_C
Mexican Egg Bake	0.55	0.55	0.55	0.67	0.67	0.67	0.61	0.61	0.61
Classic Thumbprint Cookies	0.58	0.37	0.37	0.32	0.29	0.29	0.41	0.33	0.33
Cranberry Glazed Roast Pork	0.48	0.46	0.52	0.31	0.31	0.33	0.38	0.37	0.41

Table 1: Results of the evaluation for selected cases.

use synonyms for products which were mentioned in the original recipe texts. If a product was mentioned in the text, this term must be used in the data-flow. This restriction should reduce the paraphrasing problem for the data-flow. We adapted them to our scenario. Every activity in the golden standard wf had per definition a corresponding activity in the evaluated wf.

Let T and T' be sets of activities of the wfs W and W'. W' is the golden standard wf. Each activity $t_i \in T$ has a corresponding activity $t'_i \in T'$ which are equal except for the resource and product sets. Let I_i and I'_i be resource sets and O_i and O'_i product sets for the activities t_i and t'_i. The precision for an activity $t_i \in T$ is defined as: $precision(t_i) = \frac{|I_i \cap I'_i| + |O_i \cap O'_i|}{|I_i| + |O_i|}$ The recall for a activity is defined as: $recall(t_i) = \frac{|I_i \cap I'_i| + |O_i \cap O'_i|}{|I'_i| + |O'_i|}$

This leads to the evaluation functions for a wf: $precision(W) = \frac{1}{|T|} \sum_{i=1}^{|T|} precision(t_i)$ $recall(W) = \frac{1}{|T|} \sum_{i=1}^{|T|} recall(t_i)$

The F_1 measure is defined as: $F_1(W) = 2 \frac{precision(W) * recall(W)}{precision(W) + recall(W)}$

4.2 Results

Table 1 and Table 2 show that the results for the three methods are very close. The best average precision is achieved by M. A. Table 1 shows that only for 7 cases the application of additional filter (M. B) reduces the precision compared to M. A. The best average recall is performed by M. C. We are going to interpret some of the results and examine the case of the *"Mexican Egg Bake"* recipe for which the results are equal for the three methods. This can happen, when no matching association rule is found and when no cookware item is filtered out by the stop-list. For the sample of the *"Classic Thumbprint Cookies"* recipe a wrong association rule is chosen, therefore the precision and the recall is lower for M. A & B as it is for M. A. A very interesting case is the one of *"Cranberry Glazed Roast Pork"*. There we see a drop in precision from M. A to M. B. There we see first a wrong rule is used in M. B which is corrected by M. C, which ends in a higher precision for M. C. The values of the recall might raise the question, how is it possible that a filter step produces a higher recall. Due to the fact that the filter step is preceding the association rule mining step in the pipeline, we can get a different set of association rules. If the rule mining step would precede the filter step, then indeed the results would show a higher precision but a recall which cannot be higher than before the filtering.

	P	R	F_1
M. A	**0.5127**	0.3034	0.3812
M. B	0.4828	0.3124	0.3793
M. C	0.4892	**0.3130**	**0.3817**

Table 2: Summary of the average results for the three methods for precision (P), recall (R) and F_1-Measure (F_1).

5 Discussion

The results show that the data-flow which we create has room for improvement. This evaluation is more a formative evaluation, we want to identify the promising approaches which are useful for the future work. The results of the previous section indicate that the benefit of the statistical anaphora resolution is low compared to the result which is achieved with the simple approach of M. A. Although we try to build an evaluation approach which reduces the paraphrasing problem for the control-flow, it still remains for data-flow. The measurement approach is pessimistic because it uses only lexical comparison to decide if a product is relevant or not. For example *broth* and *soup* would be rated as a mismatch even if they are semantically very close or sometimes even equal. Therefore in reality the results of M. B and M. C should be better than the measurements of the evaluation. Our intention is to develop an anaphora resolution method which does not need any ontology or other special domain knowledge. The evaluation has shown, that our approach cannot fully distinguish between an anaphora and a cooking tool. Therefore we need a list of cooking tools to differentiate between a cooking tool and an anaphora but we don't need a complex ontology. The evaluation has shown, that the application of a filter for cooking tools produced better results.

6 Related work

We are going to present related work of different research areas. The area of statistical anaphora resolution had been approached by computer linguists. Gasperin and Briscoe [18] showed a statistical approach for anaphora resolution in biomedical texts. They built a system which was based on the naive-Bayes classifier. Markert et al. [19] showed an approach that was based on the number of results of a search engine query. The queries were built with all possible antecedents for that anaphora and the anaphora themself they were embedded in sample phrases e.g. *"fruits such as apples"*. The TellMe [20] system allowed the user to define procedures trough utterances in natural language which were interpreted by the system and transformed to formal wf steps. In comparison to our system, the process of the TellMe system was interactive; the user might get feedback from the system and could specify his input.

Friedrich et al. [21] developed a system to extract a formal wf representation of a textual process description. To avoid noise, a manual preprocessing step

was applied. Our approach is capable to process textual content as it is. The work of Dufour-Lussier et al. [7] is very similar to ours. They extracted a tree representation of recipes from text. In contrast to our method, the focus was on food components which were transformed by cooking activities to other food components. The TAAABLE ontology was applied to resolve references in the text by matching sets of food components, e.g. "blueberry" and "raspberry" with "fruits". They presented in [7] an evaluation for the data-flow. Their system delivered very good result. However the test set was not representative and they were only counting the first occurrence of a product within a recipe.

7 Conclusion and future work

In this paper we presented a stream based framework for wf extraction. During the development of the different methods for anaphora resolution we experimented and tried a lot of ideas. The framework supported this work by its flexibility. A similar effect can be expected during the development of new applications for other domains. The framework allowed analysing and incrementally building wfs. We presented a wf extraction application for the domain of cooking. We presented three different anaphora resolution approaches. Two approaches used association rules which were created during an analysis of the case-base of wfs. The evaluation was performed with wfs which had a semantically correct data-flow created by a human expert. In the future we are going to develop an extraction application for a different domain.

8 Acknowledgements

This work was funded by the German Research Foundation, project number BE 1373/3-1.

References

1. Weber, R.O., Ashley, K.D., Brninghaus, S.: Textual case-based reasoning. Knowledge Engineering Review **20**(3) (2005) 255260
2. Lenz, M., Hbner, A., Kunze, M.: Textual CBR. In Lenz, M., Bartsch-Sprl, B., Burkhard, H.D., Wess, S., eds.: Case-Based Reasoning Technology From Foundations to Applications. LNAI 1400, Berlin, Springer (1998) 115–137 The original publication is available at www.springerlink.com.
3. Wiratunga, N., Koychev, I., Massie, S.: Feature selection and generalisation for retrieval of textual cases. In Funk, P., Calero, P.A.G., eds.: Advances in Case-Based Reasoning. Number 3155 in Lecture Notes in Computer Science. Springer Berlin Heidelberg (January 2004) 806–820
4. Lamontagne, L., Lapalme, G.: Textual reuse for email response. In Funk, P., Calero, P.A.G., eds.: Advances in Case-Based Reasoning. Number 3155 in Lecture Notes in Computer Science. Springer Berlin Heidelberg (January 2004) 242–256

5. Sani, S., Wiratunga, N., Massie, S., Lothian, R.: Event extraction for reasoning with text. In Agudo, B.D., Watson, I., eds.: Case-Based Reasoning Research and Development. Number 7466 in Lecture Notes in Computer Science. Springer Berlin Heidelberg (January 2012) 384–398
6. Schumacher, P., Minor, M., Walter, K., Bergmann, R.: Extraction of procedural knowledge from the web. In: Workshop Proceedings: WWW'12, Lyon, France (2012)
7. Dufour-Lussier, V., Le Ber, F., Lieber, J., Nauer, E.: Automatic case acquisition from texts for process-oriented case-based reasoning. Information Systems
8. {Workflow Management Coalition}: Workflow management coalition glossary & terminology. http://www.wfmc.org/standars/docs/TC-1011_term_glossary_v3.pdf (1999) last access 05-23-2007.
9. Jurafsky, D., Martin, J., Kehler, A., Vander Linden, K., Ward, N.: Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition. Volume 163. MIT Press (2000)
10. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: an architecture for development of robust HLT applications. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. (2002) 168175
11. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1. (2003) 423430
12. Minor, M., Schmalen, D., Bergmann, R.: XML-based representation of agile workflows. In Bichler, M., Hess, T., Krcmar, H., Lechner, U., Matthes, F., Picot, A., Speitkamp, B., Wolf, P., eds.: Multikonferenz Wirtschaftsinformatik 2008, GITO-Verlag Berlin (2008) 439–440
13. Riloff, E., Phillips, W.: An introduction to the sundance and autoslog systems. Technical report, Technical Report UUCS-04-015, School of Computing, University of Utah (2004)
14. Zhu, H.: Software Design Methodology: From Principles to Architectural Styles. Butterworth-Heinemann (March 2005)
15. Tognini-Bonelli, E.: Corpus Linguistics at Work. John Benjamins Publishing (2001)
16. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Data Engineering, 1995. Proceedings of the Eleventh International Conference on. (1995) 314
17. Kowalski, G.: Information system evaluation. In: Information Retrieval Architecture and Algorithms. Springer US (January 2011) 253–281
18. Gasperin, C., Briscoe, T.: Statistical anaphora resolution in biomedical texts. In: Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1. COLING '08, Stroudsburg, PA, USA, Association for Computational Linguistics (2008) 257264
19. Markert, K., Modjeska, N., Nissim, M.: Using the web for nominal anaphora resolution. In: EACL Workshop on the Computational Treatment of Anaphora. (2003) 3946
20. Gil, Y., Ratnakar, V., Fritz, C.: Tellme: Learning procedures from tutorial instruction. In: Proceedings of the 15th international conference on Intelligent user interfaces. (2011) 227236
21. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: Advanced Information Systems Engineering. (2011) 482496