Deriving and Combining Mixed Graphs from Regulatory Documents Based on Constraint Relations

Karolin Winter[1], Stefanie Rinderle-Ma[1,2]

[1]Faculty of Computer Science, [2]Data Science @ Uni Vienna,
University of Vienna, Vienna, Austria
{karolin.winter, stefanie.rinderle-ma}@univie.ac.at

Abstract. Extracting meaningful information from regulatory documents such as the General Data Protection Regulation (GDPR) is of utmost importance for almost any company. Existing approaches pose strict assumptions on the documents and output models containing inconsistencies or redundancies since relations within and across documents are neglected. To overcome these shortcomings, this work aims at deriving mixed graphs based on paragraph embedding as well as process discovery and combining these graphs using constraint relations such as “redundant” or “conflicting” detected by the ConRelMiner method. The approach is implemented and evaluated based on two real-world use cases: Austria’s energy use cases plus the contained process models as ground truth and the GDPR. Mixed graphs and their combinations constitute the next step towards an end-to-end solution for extracting process models from text, either from scratch or amending existing ones.

Keywords: Regulatory documents, Constraint extraction, Text mining, NLP, Process discovery

1 Introduction

Due to the tremendously increasing amount of regulatory documents the reduction of the manual effort that needs to be put into, e.g., reading and understanding these documents, becomes mandatory [19]. Lately the (semi-)automatic extraction of process model information from natural language text has gained momentum in research and practice [4, 11, 24]. Inline with this, the goals of this work are to **RG1: generate process model fragments from scratch based on regulatory documents** and **RG2: compare existing process model(s) with process model fragments derived from new regulations.**

Existing approaches [3, 6, 11] suffer from shortcomings such as the need for structured input describing processes in a sequential manner or the lack of handling noise appropriately. In reality, regulatory documents are often extensive and typically, more than one regulatory document needs to be implemented or the integration of recent ones with already existing regulatory documents must be accomplished. State-of-the-art approaches would create one process model out of each regulatory document and ignore relations and connections between parts of documents or across documents leading to models that cannot directly be employed. Moreover, as pointed out in [2], it is desirable to yield not only a description of processes but also give insights on the context of processes.

Due to these challenges, **RG1** and **RG2** cannot be realized in a one-step solution. In fact, several steps are necessary [24] including pre-processing of the input data and post-processing of the output. Moreover, we argue that a multi-step approach contributes to the understandability of the results and gives users the chance for interaction and inspection of intermediate results which would not be possible with a one-step approach. This also includes valuable insights such as paragraph characterization [26]. We opt for receiving process model information in a rather abstract representation and not directly as, for example, BPMN models. This is motivated by the fact that the process model fragments can be used in different settings and contexts and moreover, can be subject to interpretation by domain experts.

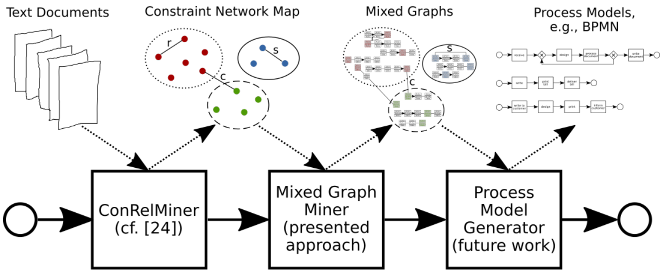


Fig. 1: Overview of the end-to-end approach

Motivated by these design choices, Figure 1 displays the multi-step approach for tackling **RG1** and **RG2** where the contribution of this work is the *Mixed Graph Miner*.

It takes as input the constraints and their relations that are determined by the ConRelMiner method [24] (step 1). The ConRelMiner enables the grouping of constraints based on, e.g., topics or different stakeholders as well as the detection of redundant, subsumed and conflicting pairs of constraints. The output of the ConRelMiner could be fed into a declarative process model, comparable to mining declarative process models from event logs, e.g., [16]. However, sentences that precede or succeed constraints can provide more detailed information on the order between the constraints resulting in process model fragments (mixed graphs) that can be used for easing the process model generation procedure. The paper at hand opts for deriving such mixed graphs and combining them with the information gained by the ConRelMiner using the underlying regulatory documents and process discovery methods. This provides an *integration of process model fragments into the overall context of the given regulatory doc-*

uments, makes the approach *robust against noise* and enables *the analysis of arbitrary as well as multiple regulatory documents*.

For the design of the Mixed Graph Miner, we state sub questions **RQ1: How to extract and present order information within paragraphs from the original text?** and **RQ2: How to put mixed graphs into relation in order to derive contextual information within and across paragraphs?**

For answering **RQ1** our approach re-embeds constraints into their context, i.e., their corresponding paragraphs and within each paragraph connections between the sentences are established which are transformed into arcs indicating control flow paths wherever suitable. The result is one mixed graph per paragraph, i.e., the document collection is represented as a set of mixed graphs. These mixed graphs can be seen as process model fragments and have to be put into relation. For **RQ2**, the additional information provided by the ConRelMiner output is used to set up connections describing relations between constraints which enables a user to recognize inconsistencies or redundancies across the document collection and to retrieve contextual information directly. The final step, i.e., the creation of, e.g., BMPN models will be tackled as future work.

The remainder of the paper is organized as follows. Section 2 outlines related work while Sect. 3 provides fundamentals on the ConRelMiner method. In Sect. 4 the contribution is described in detail and evaluated in Sect. 5. A discussion of the method is given in Sect. 6 before the paper concludes in Sect. 7 with a short summary and outlook of future work.

2 Related Work

Several existing approaches in the business process compliance domain extract information from text. The output ranges from UML models [9,17], over formal models [21], to process models. For the latter, the input varies: [12] investigate BPMN model creation from text artefacts, [6] derive BPMN models based on group stories, and [22] study the creation from use cases. [11] present an approach for BPMN process model generation from natural language text which can be seen the current state-of-the-art. Each of these approaches requires either rather structured input data (sometimes combined with additional information) or produces models that lack precision. This work takes a different approach by using extracted constraints and their relations as vehicle for extracting process models from text. Resolving relations between sentences containing constraints is not discussed in any of the mentioned approaches, but might help to improve derived business rules and process models. Related work on extracting constraints from text includes [7] which extracts SBVR rules from natural language text, but requires a domain specific model. In the information retrieval community there are several approaches targeting the improvement of techniques for deriving contextual information from natural language text. [20], for example, detects discourse and similarity across sentences. In the ConRelMiner this constitutes a part of the relation retrieval between constraints, i.e., sentences need to have a certain similarity as prerequisite. How this similarity is computed is not predefined by the approach and can be adapted to other techniques. [18] extracts short text summaries using contextual sentence information by considering surrounding sentences within paragraphs. This emphasizes the need for paragraph embedding as presented in this paper. [10] outline a method for extracting rules from legal documents by using logic-based as well as syntax-based patterns. None of these approaches aims at using constraints and text for deriving process models. [23] constructs process models from policies, but not based on text. However, the exploitation of input and output of the constraints as advocated in [23] is adopted by the work at hand.

3 Preliminaries and the ConRelMiner Method

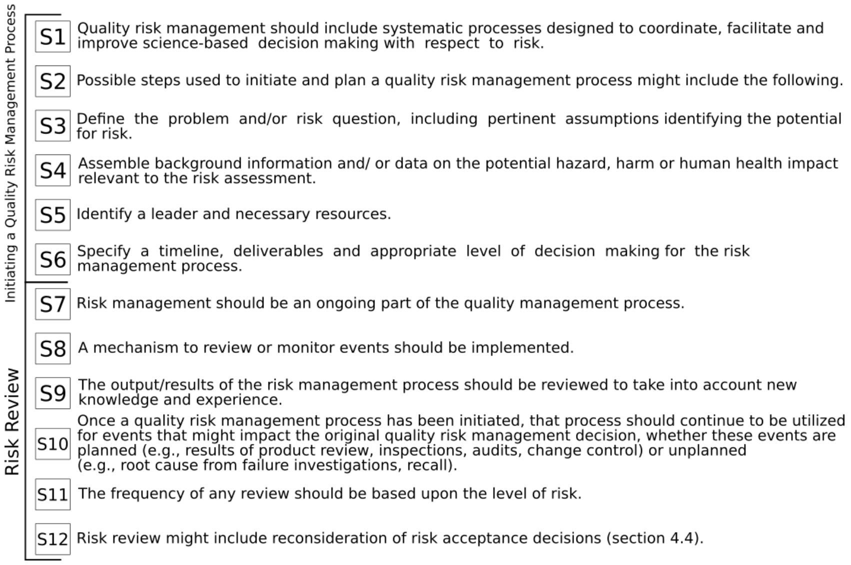
The aim of ConRelMiner (cf. [24]) is to extract constraints from a collection of regulatory documents, to group them by so-called *constraint related subjects*, e.g., topics, departments or stakeholders, and to detect three types of relations (redundancy, subsumption, conflict) between pairs of constraints. This is done by a three step approach. First of all, the documents are pre-processed. This encounters the transformation into plain text format, removing of table of contents or copyright forms, chunking into sentences and most important, extracting constraints. A constraint is defined as follows.

Definition 1 (Constraint [24]). *Let $\mathcal{S}$ be a set of sentences. A constraint is an element $s \in \mathcal{S}$ such that at least one constraint marker is contained in s. The set of all constraints is called $\mathcal{C}$.*

Constraint markers are words indicating explicit instructions, e.g., *should, shall, must*. To further illustrate this, consider a minimal real world example taken from [13] which provides guidelines for pharmaceutical quality risk management. In particular, the paragraphs displayed in Fig. 2 describe how to initiate a quality risk management process, resp. risk review. Constraints in this case are sentences *S1, S7, S8, S9, S10, S11*.

The processing step groups the elicited constraints based on constraint related subjects. This enables users to distinguish between relevant and non-relevant parts of the document collection, i.e., reduce noise. A user can choose among three different methods. The first one uses *term frequencies* and k-means++ clustering, the second one exploits the *structure of sentences* and the third one integrates *external information*, e.g., organigrams or domain knowledge provided by experts.

Afterwards, redundant, subsumed or conflicting constraint pairs are identified (cf. Def. 3 in [24]). A pair of constraints is called redundant, iff both constraints belong to the same group or the similarity of their constraint related subjects is above a user defined threshold, and their actions are similar. Subsumed constraint pairs are redundant and the action of at least the first or the second constraint contains additional information. Two constraints are called conflicting iff they belong to different groups or the similarity of constraint related


Initiating a Quality Risk Management Process
Risk Review

S1 Quality risk management should include systematic processes designed to coordinate, facilitate and improve science-based decision making with respect to risk.

S2 Possible steps used to initiate and plan a quality risk management process might include the following.

S3 Define the problem and/or risk question, including pertinent assumptions identifying the potential for risk.

S4 Assemble background information and/ or data on the potential hazard, harm or human health impact relevant to the risk assessment.

S5 Identify a leader and necessary resources.

S6 Specify a timeline, deliverables and appropriate level of decision making for the risk management process.

S7 Risk management should be an ongoing part of the quality management process.

S8 A mechanism to review or monitor events should be implemented.

S9 The output/results of the risk management process should be reviewed to take into account new knowledge and experience.

S10 Once a quality risk management process has been initiated, that process should continue to be utilized for events that might impact the original quality risk management decision, whether these events are planned (e.g., results of product review, inspections, audits, change control) or unplanned (e.g., root cause from failure investigations, recall).

S11 The frequency of any review should be based upon the level of risk.

S12 Risk review might include reconsideration of risk acceptance decisions (section 4.4).

Fig. 2: Running example – textual input

subjects is below a user defined threshold, but their two actions are similar or iff they are redundant, but contain different time spans.

The result of the ConRelMiner is a *constraint network map* $G_n = (\mathcal{C}, E)$, i.e., a graph whose nodes are constraints, $\mathcal{C}$, and whose edges, E, represent relations between constraints and are labeled as redundant, subsumed or conflicting.

Definition 2 (Constraint Network Map [24]). *A network map is a graph* $G_n = (\mathcal{C}, E)$*, with*

- $\mathcal{C}$ *being a set of nodes where each node* $c \in \mathcal{C}$ *corresponds to one constraint*
- $E \subseteq \mathcal{C} \times \mathcal{C}$ *being the edges.*

Moreover, let $w \colon E \mapsto RL := \{r, s, c\}$ *be a function assigning a label to an edge depending on the corresponding relation between the nodes that span the edge, i.e., redundant (r), subsumed (s), conflicting (c).*

In the running example using term frequencies in combination with k-means++, the constraint network map looks as depicted in Fig. 3. In this case k=3 was chosen because of the small sample size. Sentences *S1, S7, S9, S10* form the first cluster, *S8* the second one and *S11* the third one. Sentences *S7* and *S9* were detected as being subsumed.

4 Mixed Graph Miner: Deriving Mixed Graphs

The main contribution of the paper is to retrieve a mixed graph for each paragraph, which can be seen as process model fragment, (↦ RQ1) and to use the

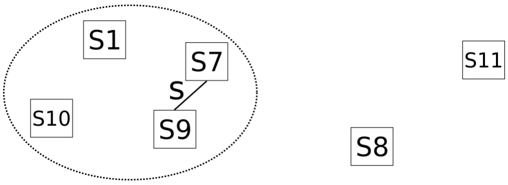


Fig. 3: Running example – output of ConRelMiner

output of the ConRelMiner for combining these graphs (↦ RQ2). Each mixed graph consists of control flow paths and, if necessary, of undirected edges. The nodes correspond to sentences which are displayed in a format that represents actors, actions and data elements. Figure 4 displays the overall method which is divided into three phases.

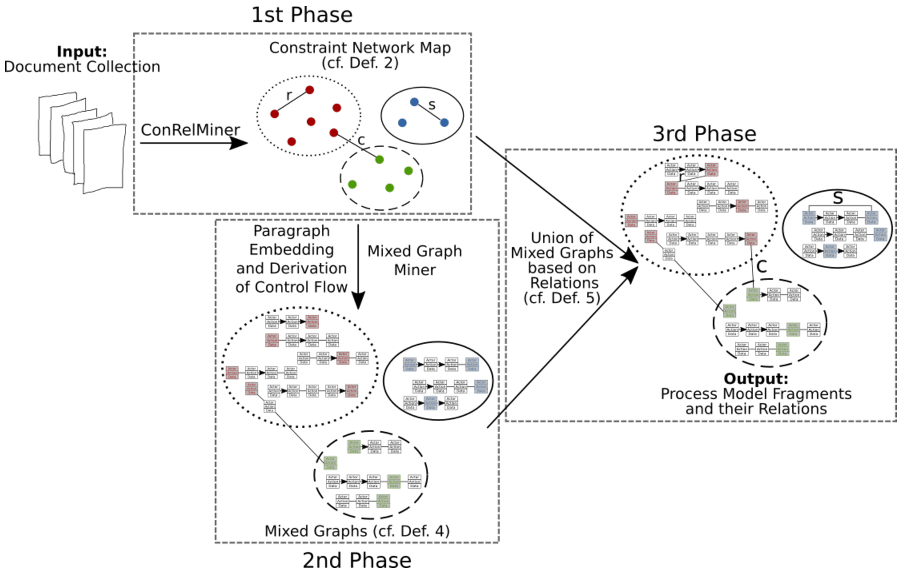


Fig. 4: Overview of method

In the first phase the prepared documents are processed by the ConRelMiner. The output consists of groups of constraints as well as a Constraint Network map describing redundant, subsumed or conflicting constraint pairs, if these are present in the collection. In the second phase, each constraint is embedded into its context, i.e., its corresponding paragraph within the original document and a mixed graph is derived for each paragraph. A user can define the granularity of paragraphs, per default, subsection level is chosen (cf. [26]). For each paragraph a set of sentences S is received which contains at least one constraint. Within each paragraph several constraints can be present. Since the aim is to resolve process fragments, each sentence within a paragraph should have a pre-and succeeding sentence except for sentences at the beginning and end of a paragraph. This corresponds to a graph-based representation and wherever possible, directed edges should be established resulting in a control flow graph (cf. Def. 3 based on [5]). We choose control flow graphs as logical representation which can be transformed into other modeling notations such as BPMN in the sequel.

Definition 3 (Control Flow Graph). *Given a set of sentences $\mathcal{S}$, a control flow graph based on $\mathcal{S}$ is defined as a directed graph $G_f = (\mathcal{S}, A)$, with*

- *$\mathcal{S}$ being a set of nodes that represent the sentences*
- *$A \subseteq \mathcal{S} \times \mathcal{S}$ being the edges that represent control flow paths.*

Control flow paths are established based on i) sequence markers (cf. [11]) and ii) input/output relations (cf. [23]). For the latter, within each sentence, the actor(s), action(s) and data must be identified and therefore each sentence is parsed to extract these types of information. As already identified by [11] sentences containing multiple clauses are a challenge. For the further processing, these are split whenever a conjunction or adverbial dependency is detected. Such dependencies can be found by traversing the parse tree provided by an NLP parser, e.g., for the running example, four subclauses in *S10* are found: **Part 1** Once a quality risk management process has been initiated; **Part 2** that process should continue to be utilized for events that might impact the original quality risk management decision, whether these events are planned; **Part 3** (e.g., results of product review, inspections, audits, change control) or **Part 4** unplanned (e.g., root cause from failure investigations, recall).

In this paper, the focus is on extracting sequences, i.e., directly follows relations. Parallel and split relations are considered as future work.

i) Using sequence markers: Markers indicating sequences are, e.g., “then”, “after”, “afterward”, “afterwards”, “subsequently”, “based on this” or “thus” (cf. [11]). Whenever such a marker is found it is checked whether it is at the beginning or at the end of a sentence. When it is at the beginning, the sentence is linked with its predecessor, otherwise it is linked to its successor. An exception is “after”, here the linking is carried out the other way round. Consider the example sentence “After a supervisor has done the assessment, the supervisor must write a report.” This sentence is split up by the parser into two subclauses. The marker would be at the beginning of the first subclause and would be linked to the predecessor of this clause, which would be wrong in this case. Considering the reordered sentence “A supervisor must write a report, after the supervisor has done the assessment.” leads to an even more complex situation, since the reordering would be wrong again because the second clause containing the “after” must now be linked to its predecessor. Therefore, it must also be considered if “after” is within a sentence containing multiple clauses or not and the linking is not just carried out based on the location of “after” within a clause but also where the clause is located in the original sentence.

ii) Using actors and data elements: Whenever no explicitly stated sequence can be found, in a second step the input/output technique of [23] is applied. Therefore process elements are derived from the text. In order to reflect process elements, each sentence in $\mathcal{S}$ is represented in a structure containing the actor, action, or data elements. This is done by exploiting the NLP tags of a sentence, i.e., an action within a sentence is represented by a verb, the actor is the subject, and data elements are viewed as objects. Challenges like resolving determiners or pronouns like "they" are already tackled during the pre-processing of documents for the ConRelMiner. Determiners or pronouns are replaced by the first preceeding subject that is found within the text and if a sentence contains multiple subjects in multiple clauses it is split into multiple smaller sentences. The ordering of the clauses is hereby maintained. Between two sentences $s_1, s_2 \in S$ a sequence $s_1 \rightarrow s_2$ is established whenever the data element of s_1 becomes the actor of s_2. This technique does not demand a sequential ordering within one paragraph. However, this technique can only be applied when each clause is self-contained, i.e., has an actor, action and data element. If this is not the case, i.e., no reasonable actor, action and data element can be found, the subclause itself is displayed.

Whenever no evidence on a control flow path between sentences can be found, an undirected edge, connecting the sentence with its predecessor and successor is integrated, leading to a so-called mixed graph (cf., e.g., [14]).

Definition 4 (Mixed Graph). *Given a control flow graph $G_f = (\mathcal{S}, A)$ on a set of sentences S, a mixed graph $G_m = (\mathcal{S}, A, E_s)$ is a graph with*

- *$\mathcal{S}$ being the set of nodes*
- *A is the set of directed edges (arcs) representing the control-flow paths between nodes*
- *E_s is the set of undirected edges, in particular, a set of tuples where each element consists of a sentence $s \in \mathcal{S}$ and its direct predecessor resp. successor.*

The usage of mixed graphs becomes necessary, since connections within a paragraph shall be established, but it cannot be guaranteed that a paragraph is described in a sequential order, i.e., that the connection is always a control flow path, which is likely to happen for real life documents (for more details, see Sect.6). Consequently, per paragraph a connected mixed graph is received which corresponds to a process model fragment. ($\mapsto$ RQ1) For reaching RQ2, i.e., the derivation of connections across paragraphs, two or more mixed graphs $\{G_m\}_{i, i \in \mathbb{N}_{>1}}$ and the constraint network graph G_n are combined into one graph (third phase of the approach).

Definition 5 (Union of Graphs based on Constraint Relations). *Let $G_n = (\mathcal{C}, E)$ be the network graph for the given document collection, $\mathcal{S}_1$ and $\mathcal{S}_2$ be two sets of sentences and $G_{m_1} = (\mathcal{S}_1, A_1, E_{s_1})$ and $G_{m_2} = (\mathcal{S}_2, A_2, E_{s_2})$ be their two mixed graphs. Let $E' \subseteq E$ such that $E' \subseteq \mathcal{S}_1 \times \mathcal{S}_2$. The union of G_{m_1} and G_{m_2} based on G_n is defined as*

$$(G_{m_1} \cup G_{m_2})_{G_n} := \begin{cases} (\mathcal{S}_1 \cup \mathcal{S}_2, A_1 \cup A_2, E_{s_1} \cup E_{s_2} \cup E') & \textit{if } E' \neq \emptyset \\ \emptyset & \textit{otherwise.} \end{cases}$$

The label of each edge in E' is preserved.

Definition 5 holds for an arbitrary number of compositions since the union $(G_{m_1} \cup G_{m_2})_{G_n}$ is again a graph.

The overall result is a set of mixed graphs that reflect one process model fragment per paragraph, contain at least one constraint and might be partly connected with each other via the relations retrieved by the ConRelMiner. Figure 5 displays the three possibilities how a redundant, subsumed or conflicting constraint pair can connect nodes of two mixed graphs.

I The constraint pair connects two nodes within the same mixed graph.
II The constraint pair connects the end and start node of two different mixed graphs.
III The constraint pair connects arbitrary nodes of two different mixed graphs.

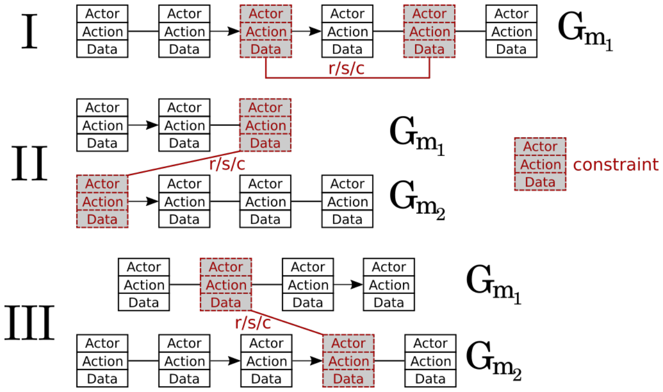


Fig. 5: Possible connections of graphs based on relations

In the case of a `redundant(r)` or `subsumed(s)` relation, case I could indicate a loop. If the relation is `conflicting(c)` a split might be present (cf. [24]) and both branches are described within the paragraph. These relations are in general not considered by state-of-the-art approaches because they are implicitly given in the text. In case II both processes could be combined directly as depicted, i.e., both paragraphs are combined using the constraint pair. In this case for redundant or subsumed relations, this might not indicate a loop but a logic connection between two paragraphs, i.e., processes. For conflicting constraint pairs the situation is like in case I but now both branches are described in separate paragraphs. Note that the paragraphs do not need bo be in a sequential order. Case III is more difficult since it is not possible to directly combine both graphs as in case II since it is unclear how to reorder the nodes before and after the related constraint pair. For conflicting constraint pairs it might mean that the two processes contradict each other because either similar actions are in the scope of different constraint related subjects or similar tasks are within the scope of the same constraint related subject but with different time spans. For redundant or subsumed constraint pairs this case is similar to case II. The reordering of nodes is in each case up to the user.

Figure 6 displays the final result for the running example. It can be seen that almost every edge is undirected in this case due to the missing markers. *S10* was split up into four different parts. The last two parts are not intended by our method, but are produced since the parse tree is searched for each conjunction as well as adverbial clause. In this case no actor, action and reasonable data element can be found and consequently the whole sentence part is returned. However, the method detects that there is a sequence indicated, i.e., directed edges are found between parts of *S10*. These mixed graphs, i.e., process model fragments can serve as input for, e.g., generating process models from scratch. The subsumed relation indicates that the two process elements (*S7* and *S9*) could be combined into one element. For the running example this seems to be only a small benefit but whenever redundancies or subsumptions are detected across several process model fragments the advantages of the approach become evident like it is demonstrated in the evaluation.

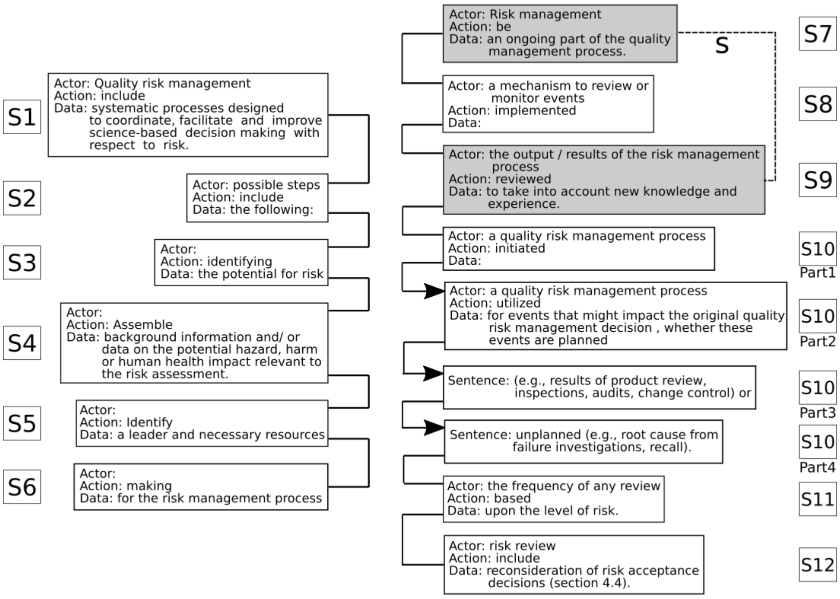

S1
Actor: Quality risk management
Action: include
Data: systematic processes designed to coordinate, facilitate and improve science-based decision making with respect to risk.
S2
Actor: possible steps
Action: include
Data: the following:
S3
Actor:
Action: identifying
Data: the potential for risk
S4
Actor:
Action: Assemble
Data: background information and/ or data on the potential hazard, harm or human health impact relevant to the risk assessment.
S5
Actor:
Action: Identify
Data: a leader and necessary resources
S6
Actor:
Action: making
Data: for the risk management process
S7
Actor: Risk management
Action: be
Data: an ongoing part of the quality management process.
S
S8
Actor: a mechanism to review or monitor events
Action: implemented
Data:
S9
Actor: the output / results of the risk management process
Action: reviewed
Data: to take into account new knowledge and experience.
S10 Part1
Actor: a quality risk management process
Action: initiated
Data:
S10 Part2
Actor: a quality risk management process
Action: utilized
Data: for events that might impact the original quality risk management decision , whether these events are planned
S10 Part3
Sentence: (e.g., results of product review, inspections, audits, change control) or
S10 Part4
Sentence: unplanned (e.g., root cause from failure investigations, recall).
S11
Actor: the frequency of any review
Action: based
Data: upon the level of risk.
S12
Actor: risk review
Action: include
Data: reconsideration of risk acceptance decisions (section 4.4).

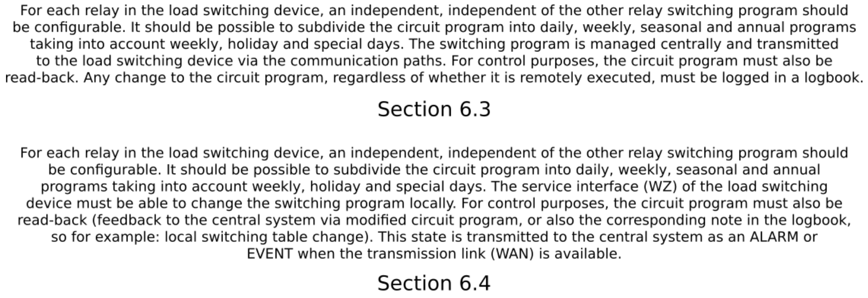
Fig. 6: Mixed Graphs – running example

5 Evaluation

The *Mixed Graph Miner* is prototypically implemented on top of the ConRelMiner using Python 3 in combination with the NLP framework Spacy (`https://spacy.io`), NLTK [8], and WordNet (`https://wordnet.princeton.edu/`). The ConRelMiner can group constraints based on three different methods. This paper opts for grouping based on *sentence structure* since this yields the highest precision according to [24]. The first case study on a regulatory document from the energy domain [1] features an end-to-end scenario for **RG1**, i.e., the derivation of a process model from scratch. The second case study on the GDPR (`https://bit.ly/2Fa05Kl`) tackles **RG1**, i.e., the comparison of existing process models with model fragments stemming from new regulations.

5.1 Austrian Smart Metering Use-Cases

Smart Metering Use-Cases for the Austrian Advanced Meter Communication System [1] contains information on processes w.r.t. smart metering and spans 91 pages. The document is written in German and was translated into English using Google Translate (`http://translate.google.com`) and a manual refinement. For some processes described within this document manually created and by experts evaluated BPMN models are available, i.e., our results can be compared to those models. The parameters for the ConRelMiner are 0.96 for the overall similarity between sentences and 0.8 for the similarity between constraint related subjects resp. tasks. Two redundant constraint pairs connect Sections 6.3 and 6.4. The text of these sections is given in Fig. 7 while the output, i.e., union of two mixed graphs, produced by the presented method is given in Fig.8. For Section 6.3 a BMPN model is available (cf. Fig. 9) which is used for comparison.



For each relay in the load switching device, an independent, independent of the other relay switching program should be configurable. It should be possible to subdivide the circuit program into daily, weekly, seasonal and annual programs taking into account weekly, holiday and special days. The switching program is managed centrally and transmitted to the load switching device via the communication paths. For control purposes, the circuit program must also be read-back. Any change to the circuit program, regardless of whether it is remotely executed, must be logged in a logbook.

Section 6.3

For each relay in the load switching device, an independent, independent of the other relay switching program should be configurable. It should be possible to subdivide the circuit program into daily, weekly, seasonal and annual programs taking into account weekly, holiday and special days. The service interface (WZ) of the load switching device must be able to change the switching program locally. For control purposes, the circuit program must also be read-back (feedback to the central system via modified circuit program, or also the corresponding note in the logbook, so for example: local switching table change). This state is transmitted to the central system as an ALARM or EVENT when the transmission link (WAN) is available.

Section 6.4

Fig. 7: Textual input – Austria's energy use cases

It can be seen that within the mixed graph no directed edges were found, i.e., no clear order is given (explicitly) in the text. The first three nodes of the graph for section 6.3 (counted from above) describe general instructions and are reflected as swimlanes by the BPMN model, the fourth node represents the first message event, the fifth node the second message event. The last node indicates the logbook entry. The actor was not resolved correctly in this case since looking at the original text, the actor should have been "any change in the circuit program". The BPMN model was checked and adapted by a domain

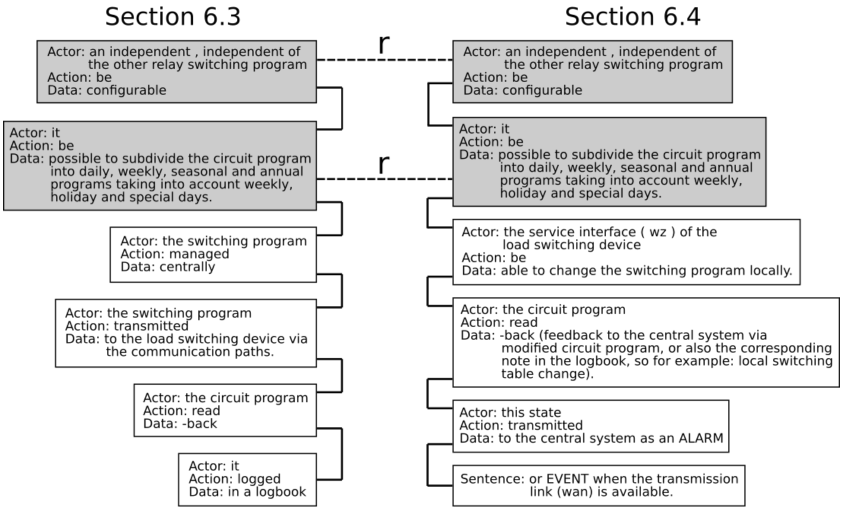

Section 6.3
Section 6.4
r
r
Actor: an independent , independent of the other relay switching program
Action: be
Data: configurable
Actor: it
Action: be
Data: possible to subdivide the circuit program into daily, weekly, seasonal and annual programs taking into account weekly, holiday and special days.
Actor: the switching program
Action: managed
Data: centrally
Actor: the switching program
Action: transmitted
Data: to the load switching device via the communication paths.
Actor: the circuit program
Action: read
Data: -back
Actor: it
Action: logged
Data: in a logbook
Actor: an independent , independent of the other relay switching program
Action: be
Data: configurable
Actor: it
Action: be
Data: possible to subdivide the circuit program into daily, weekly, seasonal and annual programs taking into account weekly, holiday and special days.
Actor: the service interface (wz) of the load switching device
Action: be
Data: able to change the switching program locally.
Actor: the circuit program
Action: read
Data: -back (feedback to the central system via modified circuit program, or also the corresponding note in the logbook, so for example: local switching table change).
Actor: this state
Action: transmitted
Data: to the central system as an ALARM
Sentence: or EVENT when the transmission link (wan) is available.

Fig. 8: Mixed graphs – Austria's energy use cases

expert. Regarding just the textual description, it remains unclear that there should be, e.g., a parallel branch without consulting an expert. Consequently, the BPMN model is ahead in terms of correctness and readability.

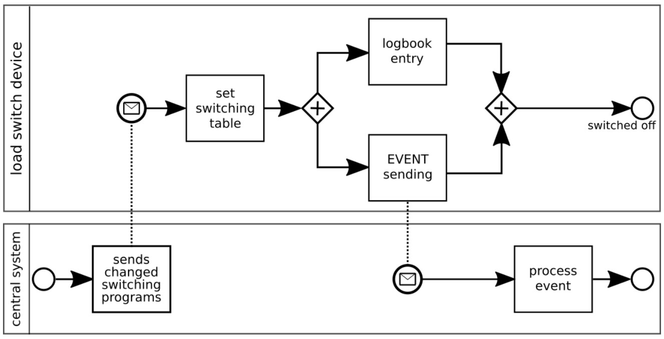


Fig. 9: BPMN model – Austria's energy use cases

However, the benefit of the presented method becomes evident when taking the relations derived by the ConRelMiner into account. With these, a similarity between both graphs is revealed since there are two redundant constraints at the beginning of each section. A user trying to create process models out of this document would be directly pointed to that fact. Even if the user decides not to join both processes, the modeling effort can be reduced since the corresponding parts could be copied. This is definitely relevant when modeling a long document from scratch like it is intended for **RG1**.

5.2 General Data Protection Regulation

In the second case study, the General Data Protection Regulation (GDPR) is fed into the method in order to demonstrate a solution for **RG2**, i.e., the integration of new regulations into already existing process models. The GDPR is a legislatory document consisting of 88 pages and was analyzed in [25], using the ConRelMiner applying the grouping method *external information* leading to promising results in terms of reduction of reading effort. In this paper, the *sentence structure* method is used with parameters 0.95 for the overall similarity of constraints and 0.8 for the constraint related subject and task similarity. The mixed graphs, i.e., process model fragments, created by the approach can be utilized to ensure compliance with the GDPR by integrating additional process tasks into already existing process models. As real-life case, process models describing procedures within the Faculty of Computer Science at the University of Vienna are used (cf. [15]). For this setting, groups referring to, e.g., *data subject* (corresponding to students) or *controller* are of interest, whereas instructions concerning, e.g., *member states* are not directly affecting the processes of the faculty and are disregarded. The approach detected, for example, constraints in Article 15 (Right of access by the data subject) and 17 (Right to erasure) as being subsumed indicating that process steps for these articles might be merged. According to experts, a centralized list containing all services that process student data was introduced and it can be applied in both cases, i.e., for granting information as well as for checking which data must be erased. Within the relevant processes, e.g., the technical staff process, an abstract additional task *delete student data* could be added. However, concrete time limits for erasure are not mentioned within the GDPR. For this task, further documents must be examined and each time limit also depends on the type of data that was stored. This is a very complex procedure and by now, a concrete process on how to erase or communicate the stored data does not exist. Our approach can deliver a structured overview of the instructions stemming from the GDPR which might help to model such processes or integrate single process steps wherever suitable.

6 Discussion and Limitations

Why is it reasonable to not just extract a mixed graph or process model per paragraph using one of the state-of-the-art approaches but to integrate the results of the ConRelMiner? The analysis of regulatory documents, especially when more than one document needs to be considered can be cumbersome. State-of-the-art approaches for extracting process models from natural language text would produce one huge model per document. However, one could argue that each document could be split into paragraphs resulting in several process model fragments. This result is received as an intermediate step by the presented approach (in this case a mixed graph per paragraph). However, connections between these graphs, i.e., the contextual information is still neglected. The presented method overcomes this issue by integrating information on relations between constraints.

What happens if no relations are found? Then each paragraph is viewed separately, which would also be the case for state-of-the-art approaches.

Why should mixed graphs be used, i.e., why is not every edge a control flow path? During the study of several regulatory documents from various domains, we realized that in most cases it can neither be assumed that a process is described sequentially within a paragraph nor be demanded that the ordering of each paragraph is sequential across the document. In contrast, mostly the description resembles an enumeration and therefore no evidence is provided in which order the steps of the process have to be carried out. In this case, it should be up to the user to decide on the order of process steps.

7 Conclusion and Future Work

Extracting process models from regulatory documents is a challenging task. First of all, regulatory documents are not necessarily structured in a process-oriented way and may contain noise. Secondly, process models are rich in information, i.e., contain orderings and may refer to different perspectives such as resources and data. In this work, we opted for constraints as vehicle to extract process fragments, represented as mixed graphs, in combination with paragraph embedding (↦ RQ1). The derived mixed graphs are put into context by exploiting relations between constraints that were extracted using the ConRelMiner method (↦ RQ2). These mixed graphs can serve as input for either process modeling from scratch or comparing and updating already existing process models. The case studies of Austria's energy use cases and the GDPR in higher education processes assess the approach as promising and illustrate how it could be used towards an end-to-end solution from text documents to process models. Future work will improve on the accuracy for deriving process elements and detect parallel and splits for generating BPMN models.

Acknowledgment

This work has been funded by the Vienna Science and Technology Fund (WWTF) through project ICT15-072.

References

1. Smart Metering Use-Cases für das Advanced Meter Communication System (AMCS), Version 1.0. Tech. Rep. 1/88, Österreichs Energie (2015)
2. Van der Aa, H., Carmona Vargas, J., Leopold, H., Mendling, J., Padró, L.: Challenges and opportunities of applying natural language processing in business process management. In: Computational Linguistics. pp. 2791–2801 (2018)
3. van der Aa, H., Leopold, H., Reijers, H.A.: Comparing textual descriptions to process models–the automatic detection of inconsistencies. Inf. Syst. 64, 447–460 (2017)
4. van der Aa, H., Leopold, H., Reijers, H.A.: Checking process compliance against natural language specifications using behavioral spaces. Inf. Syst. 78, 83–95 (2018)

5. Allen, F.E.: Control flow analysis. In: ACM Sigplan Notices. vol. 5, pp. 1–19 (1970)
6. de AR Goncalves, J.C., Santoro, F.M., Baiao, F.A.: Business process mining from group stories. In: Computer Supported Coop. Work in Design. pp. 161–166 (2009)
7. Bajwa, I.S., Lee, M.G., Bordbar, B.: SBVR business rules generation from natural language specification. In: AAAI spring symposium. pp. 2–8 (2011)
8. Bird, S., Klein, E., Loper, E.: Natural language processing with Python: analyzing text with the natural language toolkit. O'Reilly Media, Inc. (2009)
9. Deeptimahanti, D.K., Babar, M.A.: An automated tool for generating UML models from natural language requirements. In: Automated Soft. Eng. pp. 680–682 (2009)
10. Dragoni, M., Villata, S., Rizzi, W., Governatori, G.: Combining nlp approaches for rule extraction from legal documents. In: MIning and REason. with Legal texts (2016)
11. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: Advanced Information Systems Engineering. pp. 482–496 (2011)
12. Ghose, A., Koliadis, G., Chueng, A.: Process discovery from model and text artefacts. In: Services. pp. 167–174 (2007)
13. Group, I.E.W., et al.: Ich harmonized tripartite guideline, quality risk management q9. In: Techn. Req. for Registration of Pharmaceuticals for Human Use (2005)
14. Hansen, P., Kuplinsky, J., de Werra, D.: Mixed graph colorings. Mathematical Methods of Operations Research 45(1), 145–160 (1997)
15. Kabicher, S., Rinderle-Ma, S.: Human-centered process engineering based on content analysis and process view aggregation. In: Advanced Information Systems Engineering. pp. 467–481 (2011)
16. Ly, L.T., Maggi, F.M., Montali, M., Rinderle-Ma, S., van der Aalst, W.M.P.: Compliance monitoring in business processes: Functionalities, application, and tool-support. Inf. Syst. 54, 209–234 (2015)
17. More, P., Phalnikar, R.: Generating UML diagrams from natural language specifications. Applied Inf. Syst. 1(8), 19–23 (2012)
18. Ren, P., Chen, Z., Ren, Z., Wei, F., Ma, J., de Rijke, M.: Leveraging contextual sentence relations for extractive summarization using a neural attention model. In: Research and Development in Information Retrieval. pp. 95–104 (2017)
19. Riefer, M., Ternis, S.F., Thaler, T.: Mining process models from natural language text: A state-of-the-art analysis. Multikonf. Wirtschaftsinformatik pp. 9–11 (2016)
20. Saha, T.K., Joty, S., Hassan, N., Hasan, M.A.: Regularized and retrofitted models for learning sentence representation with context. In: Information and Knowledge Management. pp. 547–556 (2017)
21. Selway, M., Grossmann, G., Mayer, W., Stumptner, M.: Formalising natural language specifications using a cognitive linguistic/configuration based approach. Inf. Syst. 54, 191–208 (2015)
22. Sinha, A., Paradkar, A.: Use cases to process specifications in business process modeling notation. In: Web Services. pp. 473–480 (2010)
23. Wang, H.J., Zhao, J.L., Zhang, L.J.: Policy-driven process mapping (pdpm): Discovering process models from business policies. DSS 48(1), 267–281 (2009)
24. Winter, K., Rinderle-Ma, S.: Detecting constraints and their relations from regulatory documents using nlp techniques. In: On the Move to Meaningful Internet Systems. pp. 261–278 (2018)
25. Winter, K., Rinderle-Ma, S.: Untangling the gdpr using conrelminer. arXiv:1811.03399 (2018)
26. Winter, K., Rinderle-Ma, S., Grossmann, W., Feinerer, I., Ma, Z.: Characterizing regulatory documents and guidelines based on text mining. In: On the Move to Meaningful Internet Systems. pp. 3–20 (2017)