An Efficient Business Process Compliance Checking Approach

Jörg Becker, Philipp Bergener, Dominic Breuker, Patrick Delfmann, and Mathias Eggert

University of Muenster, ERCIS – European Research Center for Information Systems, Leonardo-Campus 3, 48149 Münster, Germany
Tel.: +49 (0) 251 8338 100
{becker,bergener,breuker,delfmann,eggert}@ercis.uni-muenster.de

Abstract. Assuring compliant business processes is an important task of business process management, which is commonly supported by the use of business process models. As every compliance rule corresponds with a typical structure, the detection of those corresponds to a pattern matching problem. More specifically, we encounter the problem of subgraph isomorphism. In this paper we propose an automatic business process compliance checking approach that relies on a subgraph isomorphism algorithm and that is suitable for process models in general. As common subgraph isomorphism is a problem that can only be solved in exponential time, we use an algorithm that simplifies the problem through pre-processing. This makes the isomorphism solvable in polynomial time. With the approach, we aim at supporting decision makers in business process compliance management.

Keywords: Compliance, Business Process Management, Pattern Matching, Subgraph Isomorphism, Efficiency.

1 Introduction

The financial crisis has demonstrated impressively how difficult it is to adhere to legal regulations and internal as well as external compliance requirements concerning business processes. The financial crisis even aggravates the complexity by introducing more and tighter regulations for financial institutions (Caldwell 2009; Caldwell, Bace and Lotto 2009; Opromolla 2009). This trend towards more regulation has also had an impact on IS research. Throughout the last years, an increasing number of approaches to solve compliance issues were published in this research area (Abdullah, Indulska and Sadiq 2009).

In this paper, we introduce a flexible model-driven business process compliance checking approach. It allows for specifying (nearly) any type of compliance rule, whose instances can be searched in any type of conceptual model. For this purpose, we make use of an efficient subgraph isomorphism (Ullmann 1976) approach. Its general applicability distinguishes our approach from existing ones. As the general subgraph isomorphism problem is only solvable in exponential processing time (Garey and Johnson 1979), according algorithms are inefficient and not suitable for large process models. Therefore, we propose reusing a subgraph isomorphism algorithm that makes use of pre-processing, making the isomorphism itself solvable in polynomial time. A precondition for consistent pattern matching results is that the process models are semantically unambiguous. Therefore, we restrict our approach to business process models that are semantically standardized.

2 Related Work

2.1 Business Process Compliance Checking

The concept of business process compliance denotes that the execution of certain processes complies with a set of regulations (Sadiq, Governatori and Namiri 2007). Kharbili et al. (2008) classify the implementation of control mechanisms in three time-dependent phases "Design-Time Compliance Checking", "Runtime Compliance Checking" and "Backward Compliance Checking". In this paper, we focus on Design-Time Compliance Checking.

It is related to modeling compliance rules and process models. Notable research in this field has been done by Governatori et al. (2008), Governatori et al. (2010), Stijin et al. (2006), Wörzberger et al. (2008), Sadiq et al. (2007), Liu et al. (2007). The approaches in this phase mainly focus on the extension or creation of modeling languages that are able to represent compliance rules or might be used for (semi-)automated compliance checking. One example is the Formal Contract Language (FCL) (Sadiq et al. 2007). Within this approach, the identified controls are transformed into control rules and classified with particular control tags. These control tags combine control rules with process models. This allows for (automatic) assignment to corresponding process elements (Sadiq et al. 2007).

Our approach differs from existing ones inasmuch it is generic and thus it abstracts from specific modeling languages. It is applicable to any modeling language that can be formalized as a graph (Harary 1969). This means that both the models and the compliance rules are regarded as graphs, where the compliance rule graph is searched in the model graph. Searching for (small) graphs in (large) graphs corresponds with the problem of subgraph isomorphism known from algorithmic graph theory (Ullmann 1976). Regarding compliance checking as a subgraph isomorphism problem allows for representing compliance rules of arbitrary structure. This aspect constitutes another difference to existing approaches.

2.2 Subgraph Isomorphism

One of the most widely known matching algorithms is Nauty, developed by Mckay (1981). This algorithm is based on group theory and computes the automorphism group of both the search graph and the subgraph which is then used to create a canonical labeling defining a node ordering. Subgraph isomorphism can be found by finding identical adjacency matrices of the two graphs in canonical form. This last verification step can be computed in O(n2) time. The construction of the canonical labeling, however, is a preprocessing step requiring exponential time in the worst case.

Messmer et al. (2000) extend this approach by proposing an algorithm for finding isomorphisms in a library of graphs. Their approach is based on a recursive decomposition of each graph into smaller subgraphs until the decomposed graphs consist of single nodes. The matching algorithm then exploits the fact that some decomposed graphs are identical and avoids redundant comparisons. In another work, the same authors include a decision tree into their algorithm that allows for matching an input graph against an entire library of graphs in O(n2) time with n being the size of the input graph (Messmer and Bunke 1999). This algorithm exploits both the facts that the underlying graphs are labeled and directed and accomplishes the subgraph search with impressive speed. While the construction of a decision tree is rather costly, the effort can be justified assuming that the processes in which patterns shall be searched do not change very often. Given this assumption, the decision tree has to be recomputed infrequently. This is why we chose this algorithm as a basis for our compliance checking approach (cf. Conte el al. (2004) for a detailed description of related algorithms).

3 Pattern Matching

3.1 Definition of Compliance Rules

In general, there are an infinite number of compliance rules depending on the actual domain and subjective perception. As such, we are not able to explain and pro¬vide an exhaustive list of identification heuristics for all of the rules. Instead, we show an exemplary excerpt which is based on the German money laundry law (ger. Geldwäschegesetz, GWG) (cf. Becker et al. (2010) for a comprehensive list of compliance rules).

Two sample compliance rules are depicted as subgraphs in Figure 1. A sequence requires a specific activity (here: "B") to be placed between two other subsequent activities (here: "A" and "C"). The corresponding subgraph is defined as GS=(VS,ES) with VS={A,B,C} and ES={(A,B),(B,C)}. Separation of duties requires two subsequent activities (here: "A" and "B") to be performed by two different people

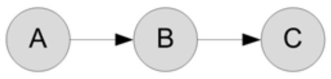
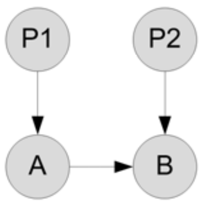
Compliance Pattern	Sequence	Separation of Duties
Corresponding Subgraph		

Fig. 1. Examples of Compliance Rules and Corresponding Subgraphs

(here: “P1” and “P2”). The corresponding subgraph is defined as GS=(VS,ES) with VS={A,B,P1,P2} and ES={(A,B),(P1,A),(P2,A)}. Different types or names of the subgraph’s vertices are realized by using a labeled graph. Unique names stored in the graph’s labels are realized through semantic standardization (cf. previous section).

3.2 The Messmer-Bunke-Algorithm for Sub-graph Isomorphism

To detect such compliance rule patterns in a library of process models we propose using an adapted version of the algorithm for subgraph isomorphism presented by MESSMER AND BUNKE in (Messmer et al. 1999). The algorithm operates on a number of a priori known model graphs (in our case the process models) and an input graph (the pattern). The basic idea of the algorithm is to compute all possible permutations of the adjacency matrices of the model graphs, with the vertex labels on the main diago¬nals of the adjacency matrices, and to arrange them in a decision tree during a pre-processing step. At runtime the adjacency matrix of the input graph is used to find identical matrices in the decision tree. The corresponding permutation matrices repre¬sent the searched subgraph isomorphism. The exact algorithm is given in (Messmer et al. 1999).

3.3 Algorithm Adaptation

There are a few modifications neces¬sary to account for particularities of our patterns. First, due to the possibility to incor¬porate wildcard elements standing for a number of arbitrary activities and prohibited activities into a pattern, we do not solve the traditional subgraph iso-morphism problem but rather a modified one in which two nodes of a pattern may be connected by a path of length k instead of being directly connected via a single edge.

To cope with this challenge we compute, for each of the model graphs, a distance matrix containing a vector of distances of all paths with maximum length k between any combinations of nodes. For each of these paths, we also associate a list of labels encountered on that path. During the creation of the decision tree, the distance matrix is treated analogously to the adjacency matrix. Thereby, for each row column element of the adjacency matrix that is associated to a node of the search graph, we also associate an appropriately permuted row column el¬ement of the distance matrix. If we now encounter a wildcard while searching the tree for a subgraph occurrence, we look up the entry of the distance matrix instead of the adjacency matrix and check if there exists a path being short enough. If the wildcard prohibits the occurrence of certain labels on the path, we additionally check if the candidate path is free of such labels.

Another problem of the original algorithm is that it solves the induced subgraph problem. That is, it requires for each edge between nodes in the model graph the presence of an edge between the associated nodes in the pattern. However, we are interested in any match of our pattern regardless of additional edges between nodes of the model graph. To adapt the algorithm in such a way that it finds every pattern, we again need to extend the decision tree. The original tree has, on each level k, a node for each permutation of an adjacency matrix of all the subgraphs of size k. Our modification is that we do want to include not only all these subgraphs but, for each of the subgraphs GS, additional subgraphs being equal to GS except that an arbitrary set of edges has been removed. When searching a pattern we can then not only find an occurrence with nodes being connected exactly as it is specified in the pattern graph but we also find any occurrence having additional edges in the model graph by matching it to one of the newly incorporated tree nodes for which exactly these edges have been removed.

A third issue with the algorithm is that we interpret the building block type assigned to a vertex as its label. However, it might be that there are certain attributes of vertices that might be of relevance when defining a pattern. Consequently, a pattern should only be returned if, besides the labels of the vertices, the corresponding attributes of the vertices also match each other. A straightforward solution to this problem would be to interpret the combination of building block type and attribute as the label of a vertex. Unfortunately, this demands that, when creating a compliance rule pattern, a value has to be assigned to all available attributes of vertices. Thus, this solution is not an option, since many patterns will leave certain attribute values unspecified. The only other option left is to apply the algorithm with labels being building block types and to subsequently check each identified occurrence of the pattern for correctness of attributes. During this post-processing step, each identified subgraph whose attributes do not match the pattern is discarded.

4 Outlook

In the short term, research will continue along two consecutive lines. We first in¬tend to implement the algorithm in order to test it on a large library of business processes. In a second step we aim at defining a large set of compliance rule patterns. With these two librar¬ies of both input graphs as well as process models we intend to conduct an empirical analysis of the algorithm's practical runtime behavior, since the algorithm has to prove its efficiency not only theoretically but also in practice. Results will point to further optimiza¬tion potential of the extended algorithm.

Although the relevance of our approach was confirmed in preliminary discussions with compliance experts in the financial sector, the support of compliance management experts through the introduced approach must be shown in detail through qualitative and quantitative studies. First evidence from accompanying research to this paper in conjunction with compliance experts suggests that the complexity of compliance management will be better handled and thus the effort for compliance checking will be lower by using the presented approach.

References

Abdullah, S.N., Indulska, M., Sadiq, S.: A Study of Compliance Management in Information Systems Research. In: ECIS, Scholar One, Verona (2009)

Becker, J., Ahrendt, C., Coners, A., Weiss, B., Winkelmann, A.: Business Rule Based Extension of a Semantic Process Modeling Language for Managing Business Process Compliance in the Financial Sector. Lecture Notes in Informatics: Modellierung betrieblicher Informationssysteme (MobIs) 175(1), 201–206 (2010)

Caldwell, F.: The Worldwide Economic Crisis will Bring Real-Time Reporting for Risk Management Gartner Research. Gartner, Inc. (2009)

Caldwell, F., Bace, J., Lotto, R.J.D.: U.S. Financial System Regulatory Overhaul Brings More Scrutiny Gartner Research. Gartner, Inc. (2009)

Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty Years of Graph Matching in Pattern Recognition. International Journal of Pattern Recognition and Artificial Intelligence 18(3), 265–298 (2004)

Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, New York (1979)

Governatori, G., Hoffmann, J., Sadiq, S., Weber, I.: Detecting regulatory compliance for business process models through semantic annotations. In: Workshop on Business Process Design, Milan (2008)

Governatori, G., Rotolo, A.: A Conceptually Rich Model of Business Process Compliance. APCCM, Brisbane (2010)

Harary, F.: Graph Theory Reading (1969)

Kharbili, M.E., de Medeiros, A., Stein, S., van Der Aalst, W.M.P.: Business Process Compliance Checking: Current State and Future Challenges. Lecture Notes in Informatics: Modellierung betrieblicher Informationssysteme (MobIs) 141, 107–113 (2008)

Liu, X., Müller, S., Xu, K.: A static compliance-checking framework for business process models. IBM Systems Journal 46(2), 335–361 (2007)

McKay, B.D.: Practical Graph Isomorphism. Congressus Numerantium: 30), 45–87 (1981)

Messmer, B.T., Bunke, H.: A Decision Tree Approach to Graph and Subgraph Isomorphism Detection. Journal of Pattern Recognition 32(12), 1979–1980 (1999)

Messmer, B.T., Bunke, H.: Efficient Subgraph Isomorphism Detection: A Decomposition Approach. IEEE Transactions on Knowledge and Data Engineering 12(2), 307–323 (2000)

Opromolla, G.: Facing the Financial Crisis: Bank of Italy's Implementing Regulation on Hedge Funds. Journal of Investment Compliance 10(2), 41–44 (2009)

Sadiq, S., Governatori, G., Namiri, K.: Modeling control objectives for business process compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 149–164. Springer, Heidelberg (2007)

Stijin, G., Vanthienen, J.: Designing Compliant Business Processes with Obligations and Permissions. In: Eder, J., Dustdar, S. (eds.) BPM Workshops 2006. LNCS, vol. 4103, pp. 5–14. Springer, Heidelberg (2006)

Ullmann, J.R.: An Algorithm for Subgraph Isomorphism. Journal of the Association for Computing Machinery 23(1), 31–42 (1976)

Wörzberger, R., Kurpick, T., Heer, T.: Checking Correctness and Compliance of Integrated Process Models. In: 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, pp. 576–583 (2008)