StarStar Models: Using Events at Database Level for Process Analysis

Alessandro Berti[1][0000−0003−1830−4013] and Wil van der Aalst[1][0000−0002−0955−6940]

Process and Data Science department, Lehrstuhl fur Informatik 9 52074 Aachen, RWTH Aachen University, Germany

Abstract. Much time in process mining projects is spent on finding and understanding data sources and extracting the event data needed. As a result, only a fraction of time is spent actually applying techniques to discover, control and predict the business process. Moreover, there is a lack of techniques to display relationships on top of databases without the need to express a complex query to get the required information. In this paper, a novel modeling technique that works on top of databases is presented. This technique is able to show a multigraph representing activities inferred from database events, connected with edges that are annotated with frequency and performance information. The representation may be the entry point to apply advanced process mining techniques that work on classic event logs, as the model provides a simple way to retrieve a classic event log from a specified piece of model. Comparison with similar techniques and an empirical evaluation are provided.

Keywords: Process Mining · Database Querying.

1 Introduction

This paper introduces StarStar models, a novel way to enable Process Mining on database events that offers the best qualities of competing techniques, providing a model representation without any effort required to the user, and offering drill-down possibilities to get a classic event log. The technique takes into account relational databases, that are often used to support information systems. Events in databases could be logged in several ways, including *redo logs* and *in-table versioning*. To retrieve an event log suitable for process mining analysis, a *case notion* (a view on the data) should be chosen, choosing specific tables and columns to be included in the event log. In order to obtain the view, a SQL query needs to be expressed and this requires a deep knowledge of the process. Moreover, this could also take to some performance issues (requiring joins between several tables). Some approaches have been introduced in literature in order to make the retrieval of event logs from databases easier: OpenSLEX meta-models [4] (this solution still requires to specify a case notion), Object-centric models [3] (where a process model is built on top of databases, but from which it's impossible to retrieve an event log) and SPARQL query translation [2]. StarStar

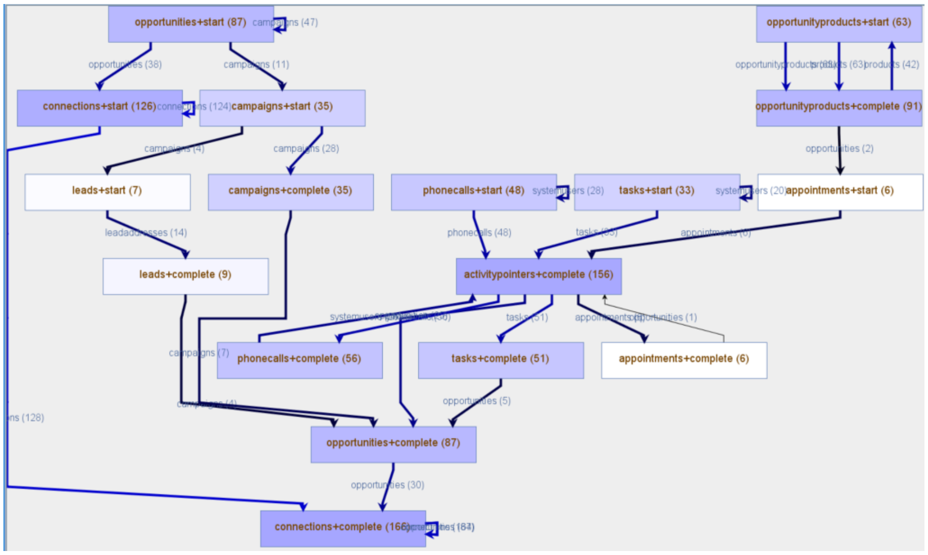

Fig. 1: Representation of a specific subset of activities in the A2A multigraph of the StarStar model extracted from a Dynamics CRM system as shown by the ProM plug-in.

models could be defined as a representation of event data contained in a database composed of several graphs: an *event to object graph* E2O that aims to represent events and objects extracted from the database and relationships between them; an *event to event multigraph* E2E that aims to represent directly-follows relationships between events in the perspective of some object; an *activities multigraph* A2A that aims to represent directly-follows relationships between activities in the perspective of some object class. StarStar models are able to display relationships between activities without forcing the user to specify a case notion, since different case notions are combined in one succint diagram. The visualization part of a StarStar model is able to show a multigraph between activities (A2A); however, relations in the E2O and E2E multigraphs are important for filtering the model and for performing a projection on the selected case notion. The E2O graph is obtained directly from the data. For the E2E and the A2A multigraphs some algorithms will be introduced in the following sections. A representation of a StarStar model extracted from a Dynamics CRM system could be found in Fig. 1.

2 Approach

StarStar models take as input an event log in a database context. In order to provide a definition of this concept (Def. 1), let $\mathcal{U}_O$ be the universe of objects, $\mathcal{U}_{OC}$ be the universe of object classes, $\mathcal{U}_A$ be the universe of activities, $\mathcal{U}_{\text{attr}}$ be the universe of attribute names, $\mathcal{U}_{\mathrm{val}}$ be the universe of attribute values. It is possible to define a function class : $\mathcal{U}_O \rightarrow \mathcal{U}_{OC}$ that associates each object to the corresponding object class.

Definition 1 (Event log in a database context) *An event log in a database context is a tuple $L_D = (E, act, attr, EO, \leq)$ where $E \subseteq \mathcal{U}_E$ is a set of events, $act \in E \rightarrow \mathcal{U}_A$ maps events onto activities, $attr \in E \rightarrow (\mathcal{U}_{attr} \not\rightarrow \mathcal{U}_{val})$ maps events onto a partial function assigning values to some attributes, $EO \subseteq E \times \mathcal{U}_O$ relates events to sets of object references, $\leq \;\; \subseteq E \times E$ defines a total order on events.*

An example attribute of an event e is the timestamp $\mathrm{attr}(e)(time)$ which refer to the time the event happened. To project an event log in a database context to a classic event log, a case notion (a set $C_D \subseteq \mathcal{P}(E) \setminus \emptyset$ such that $\bigcup_{x \in C_D} x = E$) needs to be chosen, so events that should belong to the same case can be grouped. The projection function is trivial to define, and further details could be found in [1]. The E2O graph could then be introduced:

Definition 2 (E2O graph) *Let $L_D = (E, act, attr, EO, \leq)$ be an event log in a database context. $(E \cup O, EO \subseteq E \times O)$ is an event to object graph relating events (E) and objects (O).*

The E2O graph is obtained directly from the data without any transformation. The remaining steps in the construction of a StarStar model are the construction of the E2E multigraph and of the A2A multigraph. Let $g : \mathcal{U}_O \rightarrow \mathcal{P}(\mathcal{U}_E)$, $g(o) = \{e \in \mathcal{U}_E \mid (e, o) \in \mathrm{EO}\}$ be a function that for each object returns the set of events that are related to the object, $w : \mathcal{U}_O \rightarrow \mathbb{R}$, $w(o) = \frac{1}{|g(o)|+1}$ be the weight of the object defined as the inverse of the cardinality of the set of related events to the given object plus 1, $\natural_k : \mathcal{U}_O \rightarrow \mathcal{U}_E$, $\natural_k(o) = e \quad \text{such that} \quad e \in g(o) \;\wedge\; |\{e' \in g(o) \mid e' \leq e\}| = k$ for $1 \leq k \leq |g(o)|$ be a function that in the totally ordered set $g(o)$ returns the k-th element.

Definition 3 (E2E multigraph) *Let $L_D = (E, act, attr, EO, \leq)$ be an event log in a database context. Let $F_E = \{(o, i) \mid o \in O \;\wedge\; 2 \leq i \leq |g(o)|\}$ such that for $f_E \in F_E$ the following attributes are defined: $\mathit{\Pi}^E_{obj}(f_E) \in O$ is the object associated to the edge, $\mathit{\Pi}^E_{in}(f_E) \in E$ is the input event associated to the edge, $\mathit{\Pi}^E_{out}(f_E) \in E$ is the output event associated to the edge, $\mathit{\Pi}^E_{weight}(f_E) \in \mathbb{R}^+$ associates each edge to a positive real number expressing its weight, $\mathit{\Pi}^E_{perf}(f_E) \in \mathbb{R}^+ \cup \{0\}$ associates each edge to a non-negative real number expressing its performance. For $f_E = (o, i) \in F_E$: $\mathit{\Pi}^E_{obj}(f_E) = o$, $\mathit{\Pi}^E_{in}(f_E) = \natural_{i-1}(o)$, $\mathit{\Pi}^E_{out}(f_E) = \natural_i(o)$, $\mathit{\Pi}^E_{weight}(f_E) = w(o)$,
$\mathit{\Pi}^E_{perf}(f_E) = attr(\mathit{\Pi}_{out}(f_E))(time) - attr(\mathit{\Pi}_{in}(f_E))(time)$. The event to event multigraph (E2E) can be introduced having events as nodes and associating each couple of events $(e_1, e_2) \in E \times E$ to the following set of edges: $R_E(e_1, e_2) = \big\{f_E \in F_E \mid \mathit{\Pi}^E_{in}(f_E) = e_1 \wedge \mathit{\Pi}^E_{out}(f_E) = e_2\big\}$.*

A representation of the E2E multigraph draws as many edges between a couple of events $(e_1, e_2) \in E \times E$ as the number of elements contained in the set $R_E(e_1, e_2)$.

To each edge $f_E \in R_E(e_1, e_2)$, a label could be associated in the representation taking as example the weight $\Pi^E_{\text{weight}}(f_E)$ or the performance $\Pi^E_{\text{perf}}(f_E)$.

Definition 4 (A2A multigraph) *Let $L_D = (E, act, attr, EO, \leq)$ be an event log in a database context. Let $F_A = \{(c, (a_1, a_2)) \mid c \in \mathcal{U}_{OC} \ \wedge \ (a_1, a_2) \in \mathcal{U}_A \times \mathcal{U}_A\}$ such that for $f_A \in F_A$ the following attributes are defined: $\Pi^A_{class}(f_A) \in \mathcal{U}_{OC}$ is the class associated to the edge, $\Pi^A_{in}(f_A) \in \mathcal{U}_A$ is the source activity associated to the edge, $\Pi^A_{out}(f_A) \in \mathcal{U}_A$ is the target activity associated to the edge, $\Pi^A_{count}(f_A) \in \mathbb{N}$ associates each edge to a natural number expressing the number of occurrences, $\Pi^A_{weight}(f_A) \in \mathbb{R}^+$ associates each edge to a positive real number expressing its weight, $\Pi^A_{perf}(f_A) \in \mathbb{R}^+ \cup \{0\}$ associates each edge to a non-negative real number expressing its performance. Let $AE : F_A \to \mathcal{P}(F_E)$ be a function such that for $f_a \in F_A$: $AE(f_A) = \{ \ f_E \in F_E \mid class(\Pi^E_{obj}(f_E)) = \Pi^A_{class}(f_A) \ \wedge \ act(\Pi^E_{in}(f_E)) = \Pi^A_{in}(f_A) \ \wedge \ act(\Pi^E_{out}(f_E)) = \Pi^A_{out}(f_A) \ \}$. Then for $f_A = (c, (a_1, a_2)) \in F_A$: $\Pi^A_{class}(f_A) = c$, $\Pi^A_{in}(f_A) = a_1$, $\Pi^A_{out}(f_A) = a_2$, $\Pi^A_{count}(f_A) = |AE(f_A)|$, $\Pi^A_{weight}(f_A) = \sum_{f_E \in AE(f_A)} \Pi^E_{weight}(f_E)$, $\Pi^A_{perf}(f_A) = \frac{\sum_{f_E \in AE(f_A)} \Pi^E_{perf}(f_E)}{\Pi^A_{count}(f_A)}$. The activities multigraph (A2A) can be introduced having activities as nodes and associating each couple of activities $(a_1, a_2) \in A \times A$ to the following set of edges: $R_A(a_1, a_2) = \left\{ f_A \in F_A \mid \Pi^A_{in}(f_A) = a_1 \wedge \Pi^A_{out}(f_A) = a_2 \right\}$*

A representation of the A2A multigraph (that is the visual element of a StarStar model) draws as many edges between a couple of activities $(a_1, a_2) \in A \times A$ as the number of elements contained in the set $R_A(a_1, a_2)$. To each edge $f_A \in R_A(a_1, a_2)$, a label could be associated in the representation taking as example the number of occurrences $\Pi^A_{\text{count}}(f_A)$, the weight $\Pi^A_{\text{weight}}(f_A)$ or the performance $\Pi^A_{\text{perf}}(f_A)$. Since by construction the edges in this graph can be associated to elements in the E2E graph (through the AE function), the possibility to drill down to a classic event log (choosing a case notion) is maintained. Indeed, it is possible to define a projection function from an event log in database context to a classic event log (more insights on the differences could be found in [1]) in the following way: $\text{proj}(C_D, L_D) = (C, E, \text{case\_ev}, \text{act}, \text{attr}, \leq)$ where $C = \cup_{x \in C_D} \text{id}(x)$, $\text{case\_ev} \in C \to \mathcal{P}(E)$ such that for all $c \in C_D$, $\text{case\_ev}(\text{id}(c)) = c$. A simple case notion that could be used after choosing an object class $c \in \mathcal{U}_{OC}$ is: $C_D = \cup_{o \in O, \text{class}(o) = c} \{g(o)\}$. More advanced case notions could be found in [1].

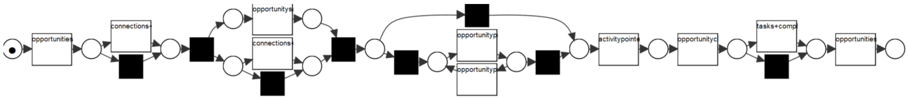

Fig. 2: Representation of the Petri net obtained choosing the *opportunity* perspective on the graph and applying projection.

An example Petri net extracted from Dynamics CRM model (the A2A multigraph has been represented in Fig. 1) could be found in Fig. 2.

3 Support tool

In order to evaluate StarStar models, a ProM plug-in has been realized that is able to take as input a representation of the events happening at database level, is able to calculate the StarStar model starting from the data and to show it to the end user using the mxGraph library. The supported input data types include XOC logs [3], that are XMLs storing events along with their related objects and the status of the object model at the time the event happened, OpenSLEX meta-models [4] and Neo4J databases. Tools for increasing/decreasing the level of complexity of the process (number of edges or number of activities) are provided. Moreover, it is provided a way to graphically filter activities/edges that are related to a given perspective. Projection functions are provided to get a classic event log out of a StarStar model when a perspective is chosen. A Petri net extracted after the projection is represented in Fig. 2.

4 Conclusions

This paper introduces StarStar models, providing a way to reduce ETL efforts on databases in order to enable process mining projects. StarStar models provide a multigraph visualization of the relationships between activities happening in a database, and the possibility to drill down. By selecting any case notion interactively we get a classic event log that can be analyzed using existing process mining techniques. Each step in the construction of a StarStar model has linear complexity and can be done on graph databases. A plug-in has been implemented on the ProM framework that can import the data, build the StarStar model, provide a visualization of the activities multigraph, and provide projection functions.

References

1. Berti, A., van Der Aalst, W.: arxiv: Starstar models: Process analysis on top of databases (2018)
2. Calvanese, D., Cogrel, B., Komla-Ebri, S., Kontchakov, R., Lanti, D., Rezk, M., Rodriguez-Muro, M., Xiao, G.: Ontop: Answering sparql queries over relational databases. Semantic Web **8**(3), 471–487 (2017)
3. Li, G., de Carvalho, R.M., van der Aalst, W.: Automatic discovery of object-centric behavioral constraint models. In: International Conference on Business Information Systems. pp. 43–58. Springer (2017)
4. de Murillas, E.G.L., Reijers, H.A., van der Aalst, W.: Connecting databases with process mining: a meta model and toolset. Software & Systems Modeling pp. 1–39 (2018)