Process Miner – A Tool for Mining Process Schemes from Event-based Data

Guido Schimm

OFFIS, Escherweg 2, 26212 Oldenburg, Germany
schimm@offis.de

Abstract. Today, process schemes are required for a lot of purposes. Extracting process schemes from event-based data is an alternative to creating them manually. Process Miner is a research prototype that can extract process schemes from event-based data. Its extracting procedure is a multistage data mining that uses a special process model. This paper outlines the main features of the tool and gives an insight into the theoretical background. Also, it describes shortly its implementation and outlines its experimental evaluation.

1 Introduction

Execution of processes is often driven by schemes. Most business processes, for example, are executed by many people together with various computer systems and possibly other tools. Here, a process schema acts as a template for process execution. It explicates a specific set of tasks, their order of execution, decision points branching to alternative executing paths, and the rules that are applied to make such decisions. In order to recognize, manage or control processes the respective schemes are needed. For example, a companies knowledge management saves and propagates process based knowledge in form of process schemes, advisory systems are based on process schemes, and workflow systems need process schemes in order to control business processes automatically.

The implicated need of process schemes requires schema development in any form. In the case that schema development is performed manually, it was realized that it is often difficult, time consuming, error prone, and expensive. An alternative to this is to generate schemes automatically. One kind of generating process schemes is extracting process schemes from event traces of properly executed processes.

Process miner is a tool that implements a complete mining procedure that extracts process schemes from event-based data. The mined schemes are complete and minimal: Complete in the sense that all recorded processes are covered by the extracted schema, minimal in the sense that only recorded processes are covered. Additionally, decision rules needed to decide between alternative paths of execution are extracted by the tool's mining procedure, too.

2 Process Miner's Major Features

The main feature of Process Miner is the process mining procedure. Mining process schemes is based on a large amount of appropriate data. Therefore, the tool expects data to be stored in a database. The user starts the procedure from a menu entry of the tool's graphical user interface. At first, Process Miner connects to a database and shows a dialog that contains a list of all process data that are in the database. The user selects the process data that should be used as input for the mining procedure. The procedure can be controlled through parameter settings. The parameters determine which mining steps should be performed, some adjustable thresholds and the levels of detail for writing the logs. After the user confirms the settings the mining procedure starts. Process Miner automatically performs the procedure as a sequence of several steps. While the mining procedure runs, the user gets a detailed log of all mining steps. The last step represents the extracted schema in a separate editor window.

Beside the mining procedure implementation Process Miner has some other noteworthy features. One of them is the schema editor. It provides two different views of a process schema at the same time. One view represents a schema as diagram, the other view shows a schema in form of a tree. Both views can be scrolled and zoomed independently. Browsing through a schema and analyzing it is very comfortable, in the case that, for example, the tree view represents an overview and the diagram view is used to show details of the schema.

The user can also edit schemes. For this purpose the editor provides functions like *add*, *delete*, *copy*, *paste*, *clone*, *move* etc. that can be applied to schema elements in both views. For example, the user can cut off parts of a mined schema and then build sub schemes or integrate them in other schemes. Also the user can model complete schemes from scratch. All schemes can be exported to other tools in form of text files in XML-format.

Another useful feature of Process Miner is its simulation component. It allows simulating processes from a particular schema in a specific context. Such a simulation context consists of a set of feature-value pairs. According to the current values alternative paths of execution inside a schema are selected by the simulation component. Altering the context of a simulation leads to different process executions. Contexts can be created by the user with an editor. Alternatively, contexts can be created automatically by Process Miner, too. During a simulation all events of starting or stopping tasks and the context states are stored into a database.

3 Implementation

Process Miner is a research prototype that is considered to be a starting point of application-specific versions. The software is designed in an object-oriented manner and implemented with Java 1.3. The current version comes in form of a single jar archive. It consists of approximately 200 classes spread over nine packages.

Process Miner is multithreaded, i.e. that all tasks are performed in separate program threads. This is important for the execution of long running mining

procedures. While such a procedure is executed the user can continue his interaction with the tool.

Process Miner comes with a customizable graphical user interface. The interface is build up in form of a multi-document interface, i.e. it consists of a desktop frame that contains and manages as many other frames as needed. The included frames are of different types. There are schema and context editor frames as well as frames for managing master data and customizing the tool. Additionally, there is a single log frame that provides commands like a simple text editor. The user interacts with the tool via a main menu, object specific pop up menus and different dialog windows.

Process Miner connects to databases via JDBC-ODBC. It can store and exchange schemes as serialized objects or in XML format.

4 Theoretical Background

Process Miner's main purpose is to extract process schemes from event-based data. This extraction can be considered a special kind of knowledge discovery from databases (KDD)[FaPS96, HaKa01]. The most specific parts of Process Miner's kind of KDD are the process model and the data mining step called process mining. Both parts were developed at the OFFIS institute of the University of Oldenburg.

The process model is a generic linear process model [Schi00]. This model defines schemes in form of nested blocks. There are two main types of blocks: Operands and operators. The tasks of a process and sub processes are operands. They are nested in operators, which build up the control flow of a particular process. The most important operators are the sequence, the parallel, and the alternative. Furthermore, the process model has an algebra. The algebra's axioms cover distributivity, associativity, and commutativity. They are the basis of a term rewriting system. This system can be used in order to transform schemes, for example, into normal forms. A normal form defines a special structure of operators.

Process mining is a multistage procedure that is based on the above process model. It can be divided into six different steps performed in sequential order.

At the first step, the procedure reads event-based data that belongs to a certain process and builds a trace for each process instance from this data. A trace is a data structure that contains all events of a process instance in alternating start and end groups that are in correct chronological order. After building the traces, they are condensed on the basis of their sequence of start and end events. Each group constitutes a path in the process schema.

At the second step, a time-forward algorithm constructs a process schema from the trace groups that is in the disjunctive normal form (DNF). A process schema in DNF starts with an alternative block and enumerates inside this block all possible paths of execution as blocks that are built up without any alternative block.

The next step deals with relations between tasks that result from the random order of performing tasks without a real precedence relation between them. These pseudo precedence relations are identified and then removed from the schema.

Because the process schema was built in DNF, it is necessary to split the schema's initial alternative and to move the partial alternatives as near as possible to the point

in time where a decision can't be retarded any longer. This is done by a transformation step. It merges blocks what also leads to a more compact schema.

It follows a decision-mining step that is based on decision tree induction [KaHa01, Quin98]. In this step an induction is performed for each decision point of the schema. The decision trees are transformed into rules and then the rules are attached to the particular alternative operators. The last step displays the extracted schema.

5 Experimental Evaluation

Up to now, the evaluation of Process Miner was based on simulated data. The tool let the user edit process schemes. These schemes can be applied for a simulation of a set of processes by Process Miner's simulation component. While it simulates processes, events of starting and ending tasks are generated and written into a database. Simulation can be parameterized and running in different contexts, so that a large amount of process variants can be generated. After simulating a process schema, a new schema can be mined from the database. If the mining procedure works accurately, the extracted schema is equal to the schema from which the data was generated.

The evaluation of Process Miner based on real world data is just in progress. First projects are in two different fields of application. One is the mining of workflow descriptions from audit logs for a workflow management system. The other should mine process schemes from production data for analytical purposes. In both cases Process Miner is supplemented by one routine that reads the special input and another routine that exports the extracted schemes in a special target format.

Literature

[FaPS96] Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth: From Data Mining to Knowledge Discovery in Databases. AI Magazine 1996.

[HaKa01] Jiawei Han, Micheline Kamber: Data Mining – Concepts and Techniques. Academic Press 2001.

[Quin98] J. Ross Qninland: C4.5 - Programs for Machine Learning. Morgan Kaufmann 1998.

[Schi00] Guido Schimm: Generic Linear Business Process Modeling. In: Stephen W. Liddle, et al. (Eds.): Conceptual Modeling for E-Business and the Web. Proceedings of the ER 2000 Workshop on Conceptual Approaches for E-Business and The World Wide Web and Conceptual Modeling. Salt Lake City, Utah, USA, 2000. LNCS 1921.