An Extensible Platform for Process Model Search and Evaluation

Christian Ress and Matthias Kunze

Hasso Plattner Institute at the University of Potsdam,
christian.ress@student.hpi.uni-potsdam.de,
matthias.kunze@hpi.uni-potsdam.de

Abstract. We present a platform that integrates a number of process model search techniques and provides a uniform interface for query formulation and search result presentation, as well as a framework to evaluate a particular search technique. The platform provides researchers with a common infrastructure to embed their search technique in form of a dedicated search engine. The demo explains the features of our platform and encourages researchers to contribute their search techniques.

1 Introduction & Background

Business process models are a central cornerstone of process-oriented organizations as they explicitly capture the knowledge to carry out the operations of a business and are reused for documentation, analysis, automation, and certification, among others. Modern companies maintain thousands of process models for reference and reuse [2], which requires effective capabilities to search among them. Researchers have proposed an abundance of techniques that focus on text, structure, or behavior of process models and allow for similarity search, i.e., obtaining approximate matches to a complete model, and querying, i.e., searching precisely by few yet relevant aspects of a model [2].

The majority of these process model search techniques, however, has been presented only theoretically. Evaluations of their quality and performance has been conducted under lab conditions, i.e., with a minimalistic implementation that mainly addresses the query processing, i.e., comparing candidate models with the query and deciding a match. This is, arguably, due to the effort of providing a complete process model search infrastructure that includes a user interface to formulate a query, a process model repository to store, manage, and retrieve models from, and the visual presentation of search results as a response to the query. This functionality is shared among all process models search techniques.

To this end, we have developed a prototypical process model search platform that assumes these tasks and allows for the integration of dedicated search techniques in form of a search engine plugin architecture. This includes a set of well-defined APIs that integrate a search engine with our platform. Moreover, the platform provides a framework to evaluate a search engine with regards to the quality of a search technique, i.e., the relevance of the provided results, and, the performance of its implementation. The platform aims to reduce the time to implement and evaluate a particular search technique, and enables the

comparison of various techniques, as they can now be deployed in the same runtime environment.

2 Search with your Search Technique

The central concept of our search platform is to enable developers to deploy a dedicated search engine to the platform and use it to search for process models in a straight-forward manner. Hence, one of the key features of our search platform is a presentation layer, which lets users specify search queries using BPMN and view ranked search results in a similar visual representation, depicted in Fig. 1.

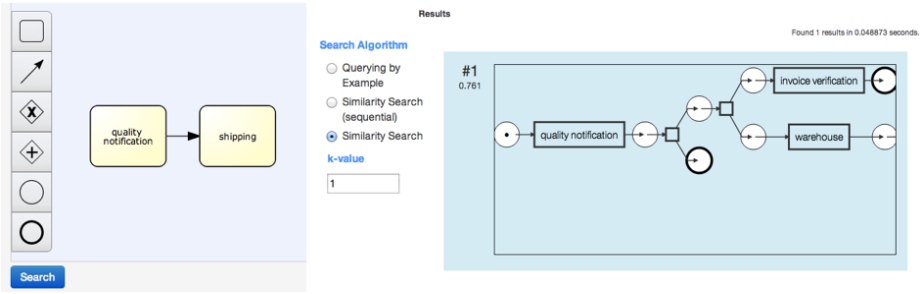

Fig. 1: Screenshot of the web-search interface.

The presentation layer includes a simplistic, web-based process model editor, that allows formulating queries as regular process models, as this is typical for similarity search and has been proposed for querying [6], too. The editor itself is highly extendable, which allows the formulation of queries in languages devised for search, e.g., BPMN-Q [1]. The input query is provided to the dedicated search engine that parses it and matches it against stored process models. To this end, BPMN queries are transformed to Petri nets.

The extensible architecture of the search platform makes it possible to integrate a dedicated search engine, provided it is implemented in Java, by providing common interfaces through which the search platform can communicate with the search engine. Currently, we require that search algorithms accept queries in form of Petri nets, defined using the jBPT Java library[1], and return search results in a similar format. We resorted to Petri nets, as they provide a common formalism for a variety of business process modeling languages [7].

Our aim is to provide researchers with an opportunity to experiment with their algorithms faster and easier, and explore the search results in an environment that is similar to one that end users expect. We do this by providing an API through which search algorithms can expose parameters that can be changed during runtime, and make these parameters accessible in the search interface. This way, parameters can be configured without modifying source code or static configurations, and without recompilation. The results of the chance are visible immediately.

[1] `http://code.google.com/p/jbpt/`

3 Evaluate your Search Technique

Our platform has originally been devised to integrate a number of dedicated search techniques in a common infrastructure. However, it turned out that the very same functionality of a search engine can be used to evaluate the underlying search technique and its implementation. That is, experiments are typically carried out by running a well-defined set of queries against a set of candidates, and assessing quality and performance. Thus, we developed a framework that allows running predefined experiments against search engines without the need for a complex evaluation infrastructure.

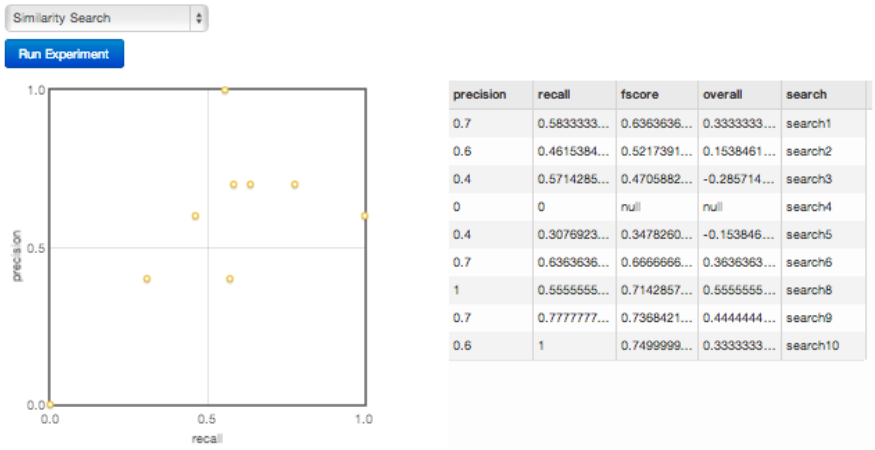

Fig. 2: Screenshot of the interface for precision/recall analysis.

Two methods for evaluating a search technique are provided. *Quality* judges on the relevance of matches in a search result with respect to a human assessment and therefore needs a reference data set. For similarity search, such a dataset has been introduced in [3]. *Performance* addresses the resource consumption of the implementation of a search engine and its scalability with regard to large process model collections. Hence, performance can be evaluated without a human data set, using any process model collection.

For this purpose, a number of time measurements and counters are provided through an API, which can be uses during the execution of a search run by a dedicated search engine. Features, such as support for laps and persistence of counters and timers over multiple search requests are available. All measurements are automatically included in the response of a query, along with statistical measures such as average, median, quantiles, standard deviation, etc.

To evaluate a search technique, we developed a web-based evaluation interface that allows choosing among a set of quality and performance evaluation methods, e.g., computation of precision and recall values and the visualization of precision-recall curves. With regards to performance measures, trend analyses can be

plotted over a number of search runs for various sizes of the candidate model collection.

Fig. 2 shows an excerpt of the evaluation interface that allows choosing a dedicated search engine and compute precision and recall values for each of the queries. The result is provided in a table for each query, and a visualization shows the queries in a coordinate system. This allows for fast identification of queries with significantly good (right upper quadrant) or poor (left lower quadrant) quality.

4 Architecture

The search platform has been implemented as a two-tiered web application, consisting of an HTML5 frontend and a Java backend, depicted in Fig. 3. The web-search and evaluation interfaces are implemented as web applications that run in common web browsers and require no installation. They communicate with the search platform via a JSON API and prove for the interaction with the user. For *search*, a simplistic process model editor based on the processWave editor[2] is provided to formulate the query. Search results are provided as an ordered list along with quality measures, cf. [4]. *Evaluation* offers predefined experiments, i.e., a set of queries and candidates, run against a dedicated search engine and visualizes results in terms of quality and performance. Particular techniques to match a query with models from the process model repository are realized in dedicated search engines.

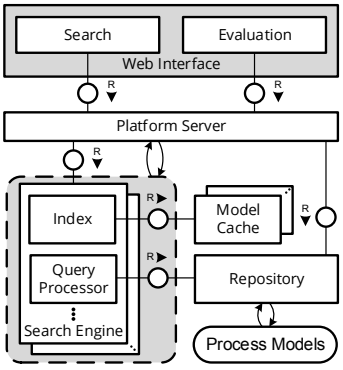

Fig. 3: Architecture of Process Model Search Platform

Search and Evaluation interfaces communicate with the platform server. As an evaluation comprises running reference searches, the platform server does not distinguish between search and experiment. That is, the experiment framework is implemented completely client-side. The search platform server integrates different dedicated *search engines*. Such an engine comprises at least a query processor that decides, whether a candidate matches the query and scores relevance. A custom index enables efficient search. To facilitate the implementation of these components, the search platform provides shared components that can be accessed by the components of a search engine, i.e., a model cache that underpins a custom index and a persistence interface with the repository that manages stored process models. The model cache increases startup speed as it preserves data that has been expensively precomputed, when models are loaded.

Through our strict use of a generic JSON API to access the search platform server it could also be used as a web service for process search and be integrated with other services or applications.

[2] `http://processwave.org`

5 Maturity & Show Case

The search platform has been implemented as a prototype to elaborate on the requirements on search engines and their integration with a common platform. Since dedicated search methods require their own query processor and indexes, such a platform provides users with a unique search interface that covers various perspectives of process model search, including similarity search and querying. At the current state, we have implemented similarity search and querying based on successor relations, cf. [5,6]. As matching is conducted on net-based formalisms, search results are currently presented using their Petri net representations. This shall be extended in future work. Also, the quality experiments are limited to human assessment of similarity, cf. [3], as other reference data was not available.

In the demo, we address researchers that are interested in process model search and may even have proposed a search technique on their own. We will introduce the platform and its architecture in a brief presentation before turning to a live demo that comprises two parts.

1. We demonstrate the search and evaluation capabilities of the platform by means of example queries, and their results. This includes a discussion of search result quality metrics and how they support users in understanding a search result. For evaluation, we show how various measures and diagrams give insight into the quality and performance of a search technique.
2. In a quick walkthrough tutorial, we explain the requirements of a custom search engine and which steps are required to integrate it into the platform by a simple example.

A screencast demonstrating our search and evaluation platform can be found at: `http://vimeo.com/ress/process-search-demo`. The platform is publicly available under the MIT open source license along with a short tutorial on how to use it and integrate a custom search engine at `http://bitbucket.org/ress/process-search`.

References

1. A. Awad. BPMN-Q: A Language to Query Business Processes. In *EMISA*, volume 119, pages 115–128, 2007.
2. R. Dijkman, M.L. Rosa, and H.A. Reijers. Managing Large Collections of Business Process Models—Current Techniques and Challenges. *Comput Ind*, 63(2):91, 2012.
3. R. Dijkman, M. Dumas, B. Dongen, R. Käärik, & J. Mendling. Similarity of Business Process Models: Metrics and Evaluation. *Inform Syst*, 36(2):498 – 516, 2011.
4. M. Guentert, M. Kunze, and M. Weske. Evaluation Measures for Similarity Search Results in Process Model Repositories. *ER '12*, pages 214–227, Springer, 2012.
5. M. Kunze, M. Weidlich, and M. Weske. Behavioral Similarity—A Proper Metric. In *BPM '11*, pages 166–181. Springer, 2011.
6. M. Kunze and M. Weske. Local Behavior Similarity. In *BPMDS '1*, volume 113 of *LNBIP*, pages 107–120. Springer, 2012.
7. N. Lohmann, E. Verbeek, and R. Dijkman. Petri Net Transformations for Business Processes—A Survey. In *Transactions on Petri Nets and Other Models of Concurrency II*, pages 46–63. Springer, 2009.