Process Analysis with bupaR 0.5.0: What’s New?

Gerhardus A. W. M. van Hulzen, Gert Janssenswillen, Niels Martin, Benoît Depaire
Research group Business Informatics
Hasselt University, Martelarenlaan 42, 3500 Hasselt, Belgium
{gerard.vanhalzen gert.janssenswillen niels.martin benoit.depaire}@uhasselt.be

Abstract—bupaR and the bupaverse are a collection of open-source R-packages designed for process data analysis in R. Due to its focus on interactivity, reproducibility, and extensibility, combined with its open-source nature, bupaR has seen a significant increase in usage over the past few years, both by academics and professional process analysts. In this demonstration, we highlight the new features of bupaR 0.5.0, which can assist practitioners when analysing their process data.

Index Terms—bupaR, R, Process analytics, Process mining, Event data

I. INTRODUCTION

Several open-source software solutions are available for process mining analyses, such as ProM [1], PM4Py [2], Apromore CE [3], and bupaR [4]. The availability of these tools allows professionals to experiment and experience the value of process mining easily and free of charge.

For process and data analysts familiar with the statistical software environment R [5], the bupaverse collection of R-packages provide a starting point for the analysis of process data. The core focus of bupaverse is based on three key principles: (i) extensibility, (ii) reproducibility, and (iii) interactivity [4] [5]. These fundamental principles, together with its open-source nature, have contributed to its widespread use.

We continuously improve and add new features to enhance the functionalities offered by bupaverse. This paper presents the release highlights of bupaR 0.5.0 [7], discusses its maturity and how one can start using it, and briefly looks forward to future development and releases.

II. NEW FEATURES

A. Activity Log

In bupaR 0.5.0, a new kind of log format has been introduced: the activity log. In an activity log, each row represents a single activity instance. This means that, as opposed to an event log in which each row represents an event occurring at a particular point in time, an activity log can have multiple timestamps per row (e.g. schedule, start, complete, etc.) [8] [9]. These are stored across multiple columns, in contrast to the single timestamp column of an event log. An example of conversion between event log to activity log and vice versa is shown in Fig. 1.

The activity log has been implemented as a new S3 class object (activitylog) alongside the existing eventlog object. The main advantages of the new activitylog object are a reduced memory footprint and increased analysis performance.

This study was supported by the Special Research Fund (BOF) of Hasselt University under Grant No. BOF190WB20.

Figure 1. Conversion from eventlog to activitylog, and v.v. [7]. Especially for analyses on activity instance level, e.g. the durations of activities, the new activitylog is more convenient and efficient because all events belonging to the same activity instance are stored on the same entry in the log. Moreover, activity attributes are recorded only once per activity instance, instead of repeatedly for each event of the same instance.

Nevertheless, this does not imply that eventlog is completely superseded. In fact, the eventlog provides more flexibility because attributes can be stored at the event level, allowing events of the same activity instance to have different attributes. For example, different resources could be responsible for the start and completion of an activity instance. In addition, in an eventlog, the same lifecycle (e.g. schedule, start, complete, etc.) can be repeated multiple times, which is useful when the activity instance was suspended and later resumed. Therefore, depending on the use case, either eventlog or activitylog is the most appropriate format. Currently, bupaR, edeaR, processmapR, and processcheckR fully support activitylog objects, and other bupaverse packages will follow in subsequent releases. Moreover, logs can be conveniently transformed from one into the other using the to_eventlog() and to_activitylog() functions.

Figure 2. bupaR S3 class inheritance schema.

In order to implement activitylog and facilitate the ex-
tendibility of the bupaR ecosystem, we have revised the S3 class inheritance of log objects. Fig. 2 visualises the new class inheritance schema. Both eventlog and activitylog are inherited from the new base log class, which in turn uses a tbl_df from the tibble package as back-end data storage. When grouping is applied to a log class using the group_by() functions, it becomes a grouped_log to signify the presence of grouping variable(s).

B. Augmenting Logs

As of edeaR 0.9.0, our package for exploratory and descriptive event data analysis, all append and append_column arguments of descriptive metrics (e.g. activity_frequency(), processing_time(), etc.) have been deprecated in favour of a new augment() method, which is consistent with the broom package for adding outputs of predictions and estimations to data. The new workflow is visualised in Fig. 3 and a code example is provided in Listing 1. For instance, we can calculate the throughput times for each case on the sepsis log and add these times back to the sepsis log as a new column "case_throughput_time".

```
sepsis %>% group_by_activity() %>% n_cases()
```

```
sepsis %>% group_by_ids(activity_id) %>% n_cases()
```

```
augment(log = sepsis, columns = "throughput_time",
      throughput_time(level = "case") %>%
      1
)
```

A new augment() method, which is consistent with the edeaR package for adding outputs of predictions and estimations to data. The new workflow is visualised in Fig. 3 and a code example is provided in Listing 1. For instance, we can calculate the throughput times for each case on the sepsis log and add these times back to the sepsis log as a new column "case_throughput_time".

```
sepsis %>%
  throughput_time(level = "case") %>%
  augment(log = sepsis, columns = "throughput_time",
          prefix = "case")
```

This new workflow ensures consistent separation between the outputs of descriptive metrics and log objects. Furthermore, the augment() method provides a standardised, flexible, and transparent way to enrich logs with descriptive metrics.

C. Improved Data Manipulation

Significant changes have been made to the supported dplyr methods for data manipulation in bupaR (e.g. filter, mutate, alias, etc.), most significantly to group_by(), for grouping event data for descriptive analyses. For example, the number of cases in which each activity was executed can be calculated using the code shown on line 1 in Listing 2.

```
sepsis %>% group_by(activity) %>% n_cases()
```

```
sepsis %>% group_by_ids(activity_id) %>% n_cases()
```

```
sepsis %>% group_by_activity() %>% n_cases()
```

A more convenient way of grouping log objects as of bupaR 0.5.0 is by using the group_by_ids() method, completed with the desired bupaR attribute function(s) (e.g. activity_id, case_id, etc.), or by directly using group_by_activity(), as shown on lines 2 and 3, respectively. These new grouping methods allow conducting grouped descriptive analyses more conveniently without knowing the underlying column names. Moreover, the handling of grouped logs is improved so that any metric can now be computed for any (set of) grouping variable(s).

III. MATURITY & USAGE

Since its conception, bupaR has received over 800K downloads in over 160 countries. Users come from various industries, e.g., healthcare, governance, automotive, and academics. Stable versions of bupaR and other bupaverse packages can be installed from CRAN using install.packages("bupaverse") or, for the version with the latest patches and bugfixes, directly from GitHub using devtools::install_github("bupaverse/bupaverse"). A demonstration of the release can be found here. Furthermore, the bupaR website contains ample documentation and examples on bupaR and the bupaverse packages.

IV. CONCLUSION & FUTURE WORK

This paper presented the release highlights of bupaR 0.5.0, most notably the introduction of the activity log, a new standardised way to augment logs, and improved data manipulation.

Future releases will focus on extending the bupaverse ecosystem with new functionalities for process analysis and maintenance of existing code. New functionalities, such as Performance Spectrum, trace and activity clustering, social network mining and process discovery, are currently on the roadmap. Other functionalities can be requested using GitHub Issues.

ACKNOWLEDGMENT

The authors would like to warmly thank all users who are actively contributing to the bupaR-framework by submitting issues and pull requests on the GitHub repositories.

REFERENCES


https://github.com/bupaverse
https://tinyurl.com/icpmdemo1