From Time Series to Process Model Forecasting

Johannes De Smedt[1], Artem Polyvyanyy[2], and Jochen De Weerdt[1]

[1] Department of Decision Sciences and Information Management
Faculty of Economics and Business, KU Leuven, Leuven, Belgium
{johannes.desmedt;jochen.deweerdt}@kuleuven.be
[2] School of Computing and Information Systems
The University of Melbourne, Australia
artem.polyvyanyy@unimelb.edu.au

Abstract. The surge in event-based data recorded during the execution of business processes is ever-growing and has spurred an array of process analytics techniques to support and improve information systems. A major strand of process analytics encompasses forecasting the process's future development, mostly focusing on next-step, remaining time, or goal-oriented predictions. The granularity of such approaches lies with the events in the process. This work approaches process forecasting at the level of the entire process model. This allows process analysts to act on predicted global and longer-term changes such as impending drifts in the process model rather than fine-granular ones such as next-step prediction. To this purpose, event data is captured at various intervals and aggregated in directly-follows graphs. The relation of each activity pair in the graph is monitored over these intervals, and their future values forecasted using standard time-series techniques. Experiments confirm that this approach is already well-capable of informing process analysts about the future status of the process.

Keywords: Process model forecasting, predictive process modelling, process mining, time series analysis

1 Introduction

Process mining...

Various predictive techniques exist in the context of process mining, many coined under the term of predictive process monitoring [[18, 19]...]...

The approach presented in this paper entails a paradigm shift from these typical predictive applications which span remaining time, next-step, or goal-oriented prediction [4] as the focus is not on the level of single outcomes at the level of the trace and/or event but on the model. These predictive techniques construct a predictive model to infer the future development of a trace, while the proposed forecasting approach infers a model which can generate future traces itself. This is achieved by an aggregation at the model level both at the input as well as the output of the learning process. This is done by using directly-follows relations over all activity pairs as a summary of the process model at various time

steps, which are further extrapolated into the future using time series techniques. An initial attempt to set up process model forecasting was presented in [3], however, the level of granularity of the forecasted horizons is coarser compared to the higher-frequency intervals used for time series underpinning the proposed approach.

Where process mining focuses on learning the as-is model to inform the to-be model and suggest potential repairs and improvements, process model forecasting allows to already grasp the future outcomes of the current as-is process which allows to shortcut potentially wrong outcomes [14].

Next to the forecasting aspect, we also present how to visualize the forecasted models and show how they exist of adaptations to the current as-is model with a level of uncertainty through change graphs [8].

This paper makes three contributions:

1. Proposes the first process model forecasting technique, which fits the the broader area of predictive process monitoring;
2. Evaluates the quality of the forecasted process models using standard conformance checking techniques from process mining;
3. Confirms that the achieved quality of forecasts is useful for practical applications.

To this purpose, a variety of time series analysis techniques are applied to 3 real-life event logs with two event log aggregations that establish suitable time series. They predict the level of directly-follows occurrences for activity pairs over time. A variety of considerations regarding the use of time series analysis for event logs are discussed, as well as the suitability of other predictive algorithms. Results show that simple forecasting techniques often already perform adequately without extensive parameter tuning. More intricate models often fail to report consistent performance.

This paper is structured as follows...

2 Preliminaries

An event log L contains the recording of traces $\sigma \in L$ produced by an information system during its execution and contains a sequence of events. Events in these traces are part of the power set over the alphabet of activities Σ which exist in the information system $\langle e_1, ..., e_{|\sigma|} \rangle \subseteq \Sigma^*$. Directly follows relations between activities in an event log can be expressed as a counting function over activity pairs $>_L$: $\Sigma \times \Sigma \rightarrow \mathbb{N}$ with $>_L (a_1, a_2) = |\{e_n = a_1, e_{n+1} = a_2, \forall e_i \in L\}|$. Directly follows (DF) relations can be calculated on traces and subtraces in a similar fashion. A Directly Follows Graph (DFG) of the process then exists as the weighted directed graph with the activities as nodes and their DF relations as weights $DFG = (\Sigma, >_L)$.

In order to obtain predictions regarding the evolution of the DFG we construct DFGs for subsets of the log. Many aggregations and bucketing techniques

exist for next-step and goal-oriented outcome prediction [18, 19], e.g., predictions at a point in the process rely on prefixes of a certain length, or particular state aggregations [1]. In the proposed forecasting approach, however, not cross-sectional but time series data will be used. Hence, the evolution of the DFGs will be monitored over intervals of the log where multiple aggregations are possible:

- Equitemporal aggregation: each sublog contains a part of the event log of equal time duration. This can lead to sparsely populated sublogs when the events' occurrences are not uniformly spread over time, however, is easy to apply (on new traces).
- Equisized aggregation: each sublog contains a part of the event log of similar DF sum. This leads to well-populated sublogs, however, might be harder to apply when new data does not contain sufficient new DF occurrences.

Time series can be obtained for all $>_{Ls}$, $Ls \subseteq L$ by applying the aforementioned aggregations. Tables 1 and 2 provide an example of both.

Case ID	Activity	Timestamp
1	A1	11:30
1	A2	11:45
1	A1	12:10
1	A2	12:15
2	A1	11:40
2	A1	11:55
3	A1	12:20
3	A2	12:40
3	A2	12:45

Table 1: Example event log with 3 traces and 2 activities.

DF	Equitemporal	Equisized
$<_{Ls}(A1, A1)$	(0,1,0)	(1,0,0)
$<_{Ls}(A1, A2)$	(1,1,1)	(1,1,1)
$<_{Ls}(A2, A1)$	(0,1,0)	(0,1,0)
$<_{Ls}(A2, A2)$	(0,0,1)	(0,0,1)

Table 2: An example of using an interval of 3 used for equitemporal aggregation (75 minutes in 3 intervals of 25 minutes) and equisized intervals of size 2 (6 DFs over 3 intervals)).

3 Methodology

This section outlines the forecasting techniques that will be used, as well as their connection with time series extracted from event logs.

3.1 Time series techniques

To model the time series of DFGs, various algorithms are used. In time series modelling, the main objective is to obtain a forecast or prediction $\hat{y}_{T+h|T}$ for a horizon $h \in \mathbb{N}$ based on previous T values in the series $(y_1, ..., y_T)$ [7]. A wide array of time series techniques exist, ranging from simple models such as

naive forecasts over to more advanced approaches such as exponential smoothing and auto-regressive models. Many also exist in a seasonal variant due to their application in contexts such as sales forecasting. Below, the most-widely used techniques are formalised.

The naive forecast simply uses the last value of the time series T as its prediction:

$$\hat{y}_{T+h|T} = y_T$$

A Simple Exponential Smoothing (SES) model uses a weighted average of past values where their importance exponentially decays as they are further into the past:

$$\hat{y}_{T+h|T} = \alpha y_T + (1-\alpha)\hat{y}_{T|T-1}$$

with $\alpha \in [0, 1]$ a smoothing parameter where lower values of α allow for focusing on the more recent values more. Holt's models introduce a trend in the forecast, meaning the forecast is not flat:

$$\hat{y}_{T+h|T} = l_t + hb_t$$

with $l_t = \alpha y_t + (1-\alpha)(l_{t-1} + b_{t-1})$ modelling the overall level of the time series and $b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1}$ modelling the trend over the series where α is as before and β the smoothing parameter for the trend. Exponential smoothing models often perform very well despite their simple setup.

AutoRegressive Integrating Moving Average (ARIMA) models are based on auto-correlations within time series. They combine auto-regressions with a moving average over error terms.

An AutoRegressive (AR) model of order p uses the past p values in the time series and applies a regression over them. It can be written as:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t$$

where all ϕ_i, $i \in [1, p]$ can assign different weights to different time lags.

A Moving Average (MA) model of order q can be written as:

$$y_t = c + \epsilon_t + \phi_1 \epsilon_{t-1} + \cdots + \phi_q \epsilon_{t-q}$$

with $\phi > 0$ a smoothing parameter, and ϵ_t again a white noise series, hence the model creates a moving average of the past forecast errors. Given the necessity of using a white noise series for AR and MA models, data is often differenced to obtain such series.

ARIMA models then combine both AR and MA models where the integration is taking place after modelling as these models are fitted over differenced time series. An ARIMA(p, d, q) model can be written as:

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_t$$

with y'_t a differenced series of order d. Forecasting is possible by introducing $T+h$ to the equation and iteratively obtaining results by starting off with $T+1$.

ARIMA models are considered to be one of the strongest time series modelling techniques.

An extension to ARIMA which is widely used in econometrics exists in (Generalized) AutoRegressive Conditional Heteroskedasticity ((G)ARCH) models [5]. They resolve the assumption that the variance of the error term has to be equal over time, but rather model this variance as a function of the previous error term. For AR-models, this leads to the use of ARCH-models, while for ARMA models GARCH-models are used as follows.

An ARCH(q) model captures the change in variance over time as follows:

$$\sigma_t^2 = \alpha_0 + \alpha_1 y_{t-1}^2 + \cdots + \alpha_q y_{t-q}^2$$

is the variance over time with $\alpha_0 > 0$ and $\alpha_i \geq 0$, $i \in [1, q]$ smoothing parameters, and $y_t = \sigma_t \epsilon_t$ where $\epsilon_t \overset{i.i.d}{\sim} (\mu = 0, \sigma^2 = 1)$. This can be used to capture that the variance gradually increases over time, or has short bursts of increased variance.

A GARCH(p,q) model combines both the past values of observations as well as the past values of variance:

$$\sigma_t^2 = \alpha_0 + \alpha_1 y_{t-1}^2 + \cdots + \alpha_q y_{t-q}^2 + \cdots + \beta_1 \sigma_{t-1}^2 + \cdots + \beta_p \sigma_{t-p}^2$$

with $\alpha_0 > 0$ and $\alpha_i, \beta_i \geq 0$, $i > 0$.

(G)ARCH models often outperform ARIMA models in contexts such as the prediction of financial indicators of which the variance often changes over time [5].

3.2 Forecasting Directly Follows series

The trade-off exists between approaching DFGs as a multivariate collection of DF time series, or treating each DF separately. The aforementioned time series techniques all use univariate data in contrast with, e.g., Vector AutoRegression (VAR) models, or machine learning-based methods such as neural networks or random forest regressors. Despite their simple setup, it is debated whether traditional statistical approaches are necessarily outperformed by machine learning methods. [12] found that this is not the case on a large number of datasets and note that the machine learning algorithms require significantly more computational power, a result that was later reaffirmed although it is noted that hybrid solutions are effective [13]. Especially for longer horizons traditional time series approaches still outperform machine learning-based models. Given the potentially high number of DF pairs in a process' DFG, the proposed approach uses a time series algorithm for each DF series separately. VAR models would require a high number of intervals (at least as many as there are DFs times the lag coefficient) to be able to estimate all parameters of such a high number of time series despite their potentially strong performance [20]. Machine learning models could potentially leverage interrelations between the different DFs but again would require long training times to account for dimensionality issues due to the potentially high number of DFs.

4 Experimental evaluation

In this section, an experimental evaluation over X real-life event logs is reported.

4.1 Re-sampling and test setup

Time series are obtained by specifying a number of intervals (i.e. time steps in the DF series) using either equitemporal or equisize aggregation. Time series algorithms are parametric and sensitive to sample size requirements [6]. Depending on the number of parameters a model uses, a minimum size of at least 50 steps is not uncommon, although typically model performance should be monitored at a varying number of steps. In the experimental evaluation, three different results will be given to study the impact of the number of steps. 100 intervals are established and time series models are trained over the first quarter, the first two quarters, and first three quarters of data points to forecast the second, third, and fourth quarter of data points respectively. This allows to both inspect the difference in result when only few data points are used, and whether there is a difference forecasting data points in the middle or towards the end of the trace.

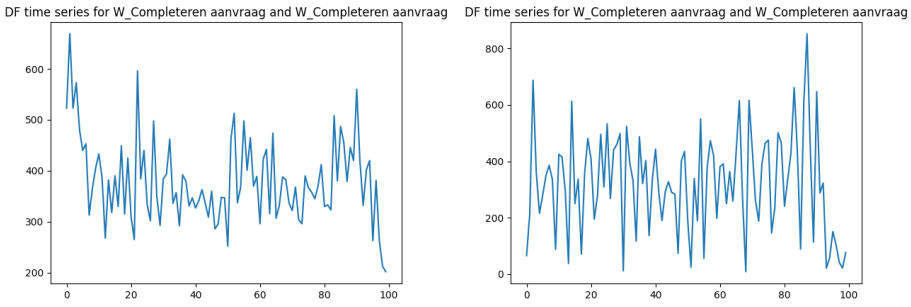
Three widely-used event logs are used: the 2012 BPI challenge log[3], the Sepsis cases event log[4], and the Road Traffic Fine Management Process log[5] (RTFMP) event log. Each of these logs has a diverse set of characteristics in terms of case and activity volume, as well as average trace length as can be seen in Table 3.

Event log	# cases	# activities	Average trace length
BPI 12	13,087	36	20.020
Sepsis	1,050	16	14.490
RTFMP	150,370	11	3.734

Table 3: Overview of the characteristics of the event logs used in the experimental evaluation.

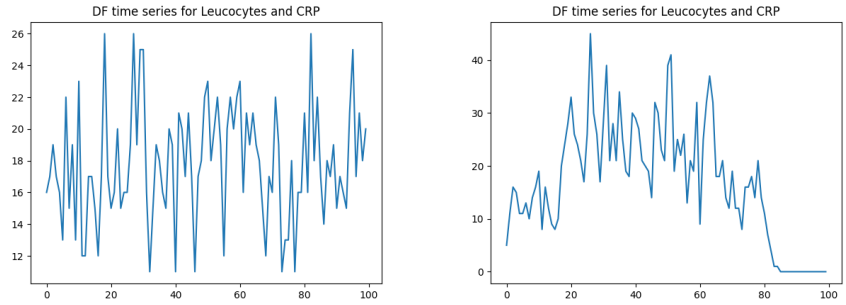
An example of applying the equisize or equitemporal aggregation to the event logs with 100 intervals results in the DF time series of Figures 1 to 3 where the most frequently occurring activity pairs are included. The equitemporal aggregation is based on a number of intervals over the whole time span of the event log. When using this aggregation, a noticeable decline of DF pairs is visible towards the end of the series. This phenomenon is typical in event logs, as processes typically have particular endpoint activity, e.g., the closure of a loan application event in the BPI 12 log. The use of a cutoff of 50 for the most-frequently occurring DF pairs results in different time series as well. For the BPI 12 log, the

[3]https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f
[4]https://doi.org/10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460
[5]https://doi.org/10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5



(a) Most common DF - equisize (b) Most common DF - equitemp

Fig. 1: BPI 12

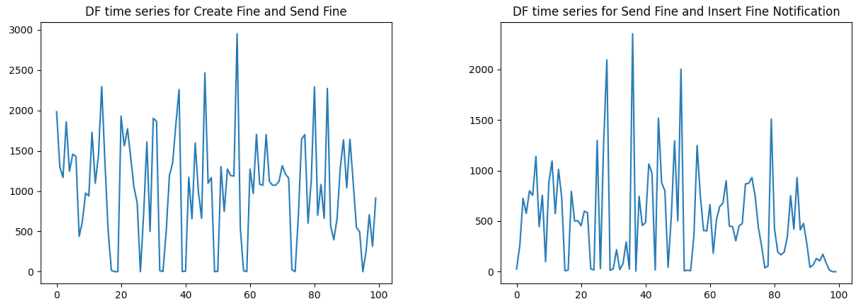


(a) Most common DF - equisize (b) Most common DF - equitemp

Fig. 2: Sepsis

frequency of the 50th pair is still relatively high, while for the other event logs the frequency of the DF pair is low and close to 0, making the series unsuitable for analysis with white noise series analysis techniques that assume stationarity. Ideally, every time series is tested using a stationarity test such as the Dickey-Fuller unit root test [11] and an appropriate lag order is established for differencing. Furthermore for each algorithm, especially ARIMA-based models, (partial) auto-correlation could establish the ideal p and q parameters. However, for the sake of simplicity and to avoid tedious solutions where each activity pair has to have different parameters, various values are used for p, d, and q and applied to all DF pairs where only the best-performing are reported below for comparison with the other time series techniques.

Resampling is based on a 10-fold cross-validation constructed following a rolling window approach for various horizon values $h \in [1, 5]$ where a recursive strategy is used to iteratively obtain $\hat{y}_{t+h|T_{t+h-1}}$ with $(y_1, \ldots, y_T, \ldots, \hat{y}_{t+h-1})$ [21]. The 10 training sets exist from $(y_1, \ldots, y_{T-h-f})$ and the test sets from



(a) Most common DF - equisize (b) Most common DF - equitemp

Fig. 3: RTFMP

$(y_{T-h-f+1}, \ldots, y_{T-f})$ with $f \in [0, 9]$ the fold index [2]. While direct strategies with a separate model for every value of h can be used as well and avoid the accumulation of error, they do not take into account statistical dependencies for subsequent predictions.

Given that we want to evaluate the capability of the approach to accurately predict the evolution of the process model, the combination of all DF predictions to obtain a global DFG prediction is considered. The following two criteria are used:

- Cosine distance: measures the distance between two vectors and is often used to compare graph distance. This metric is used to compare the DFG's edge weight matrices between the actual and predicted number of DF relations.
- Entropic relevance: a measure for stochastic conformance checking computed as the average number of bits required to compress each of the log's traces based on the structure and information about relative likelihoods provided by the model [15].

These criteria balance a predictive and structural evaluation of the algorithms and report on both the numeric performance common in a forecasting setting as well as their appropriateness in terms of reproducing a structurally usable process model which allows for the observed process behavior.

4.2 Results

All pre-processing was done in Python with a combination of *pm4py*[6] and the *statsmodels* package [16]. The code is available

The results are displayed in Figures 4 to 9.

[6]https://pm4py.fit.fraunhofer.de

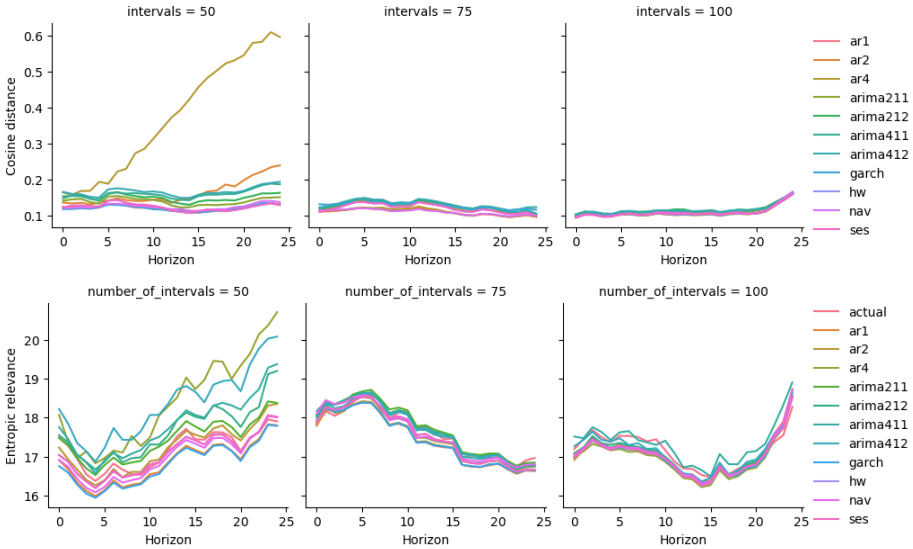


Fig. 4: Results for equi-size aggregation for the BPI12 event log.

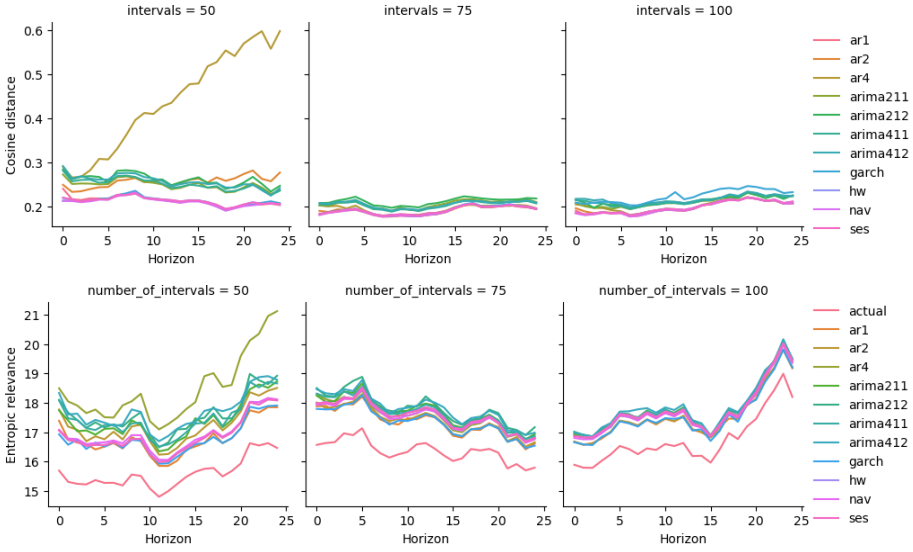


Fig. 5: Results for equi-size aggregation for the sepsis event log.

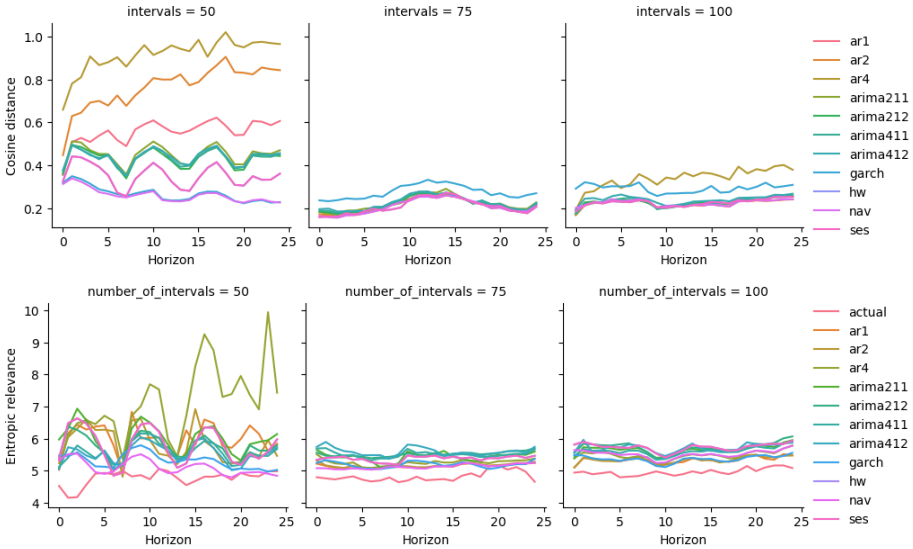


Fig. 6: Results for equi-size aggregation for the RTFMP event log.

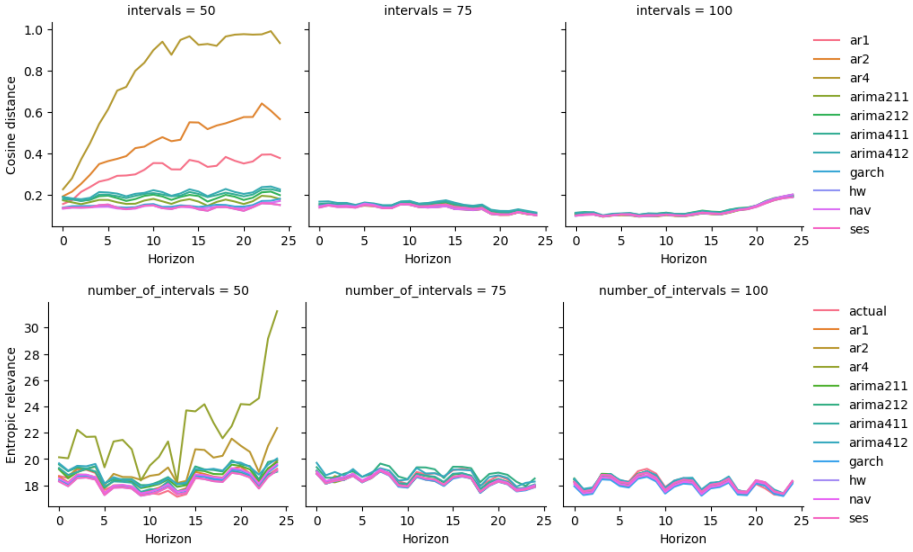


Fig. 7: Results for equi-temporal aggregation for the BPI12 event log.

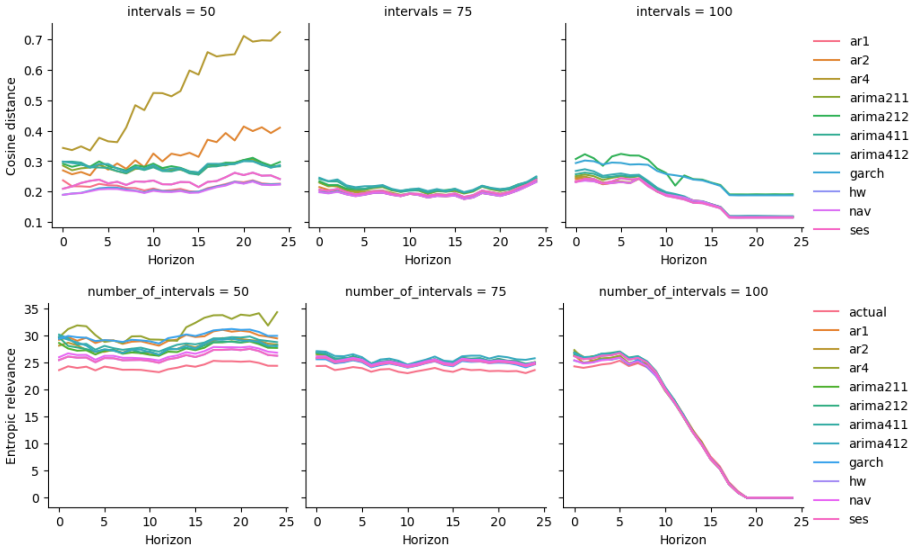


Fig. 8: Results for equi-temporal aggregation for the sepsis event log.

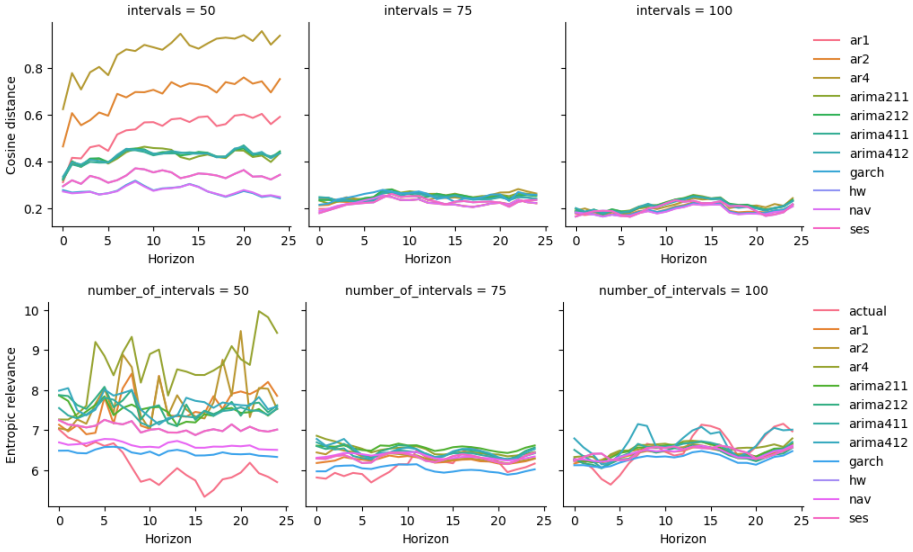


Fig. 9: Results for equi-temporal aggregation for the RTFMP event log.

5 Visualising Process Model Forecasts

The evaluation Section 4 has shown a good performance of the prediction algorithms for the process model forecasts. To that end, deriving insights from such predicted data remains a difficult task for the analyst. In this section we present the design of visualization system that aids analysts in exploration of the past and future of the processes.

We designed a Process Change Exploration (PCE) system to support the interpretation of the process model forecasts. In order to design the system we first established requirements for the visualization based on the related literature and experience working with event sequence data. We then designed several prototypes, that after several rounds of discussions matured into the implemented visualization system.

To derive the requirements we focus on the requirements of process mining analysis with respect to process forecasting and visualization principles. The authors of [14] discuss the opportunities for process forecasting. They describe that the possible utility of the process forecasting is an understanding the incremental changes or adaptations that happen to the process model into the future. In designing an exploitative visualization system, we followed the "Visual Information-Seeking Mantra:" *overview first, zoom and filter, then details-on-demand* [17].

Thus, we expect our system to assist in:

R1. **Identify process adaptations:** The visualisation system should assist the user in identifying the changes changes that happen in the process model;
R2. **Allow for interactive exploration:** The user should be able to follow the visual information-seeking principles;

User interface The Figure 10 displays the screenshot of the PCE visualization system. We improve upon the notion of Directly-Follows graph [10] that is widely used in process mining research and practice. We use the ideas from the version graph [9] on how to represent the change between versions of the graph with coloring of the transitions.

..... and so on

The interactive visualization system was implemented in D3.JS JavaScript visualization library.

6 Discussion

- Explain the potential applications;
- Indicate the managerial implications;
- What profile is needed from the time distribution of events to apply this technique? Cutting part of the trace necessary? Where to situate the predictions and/or training/test sets?;

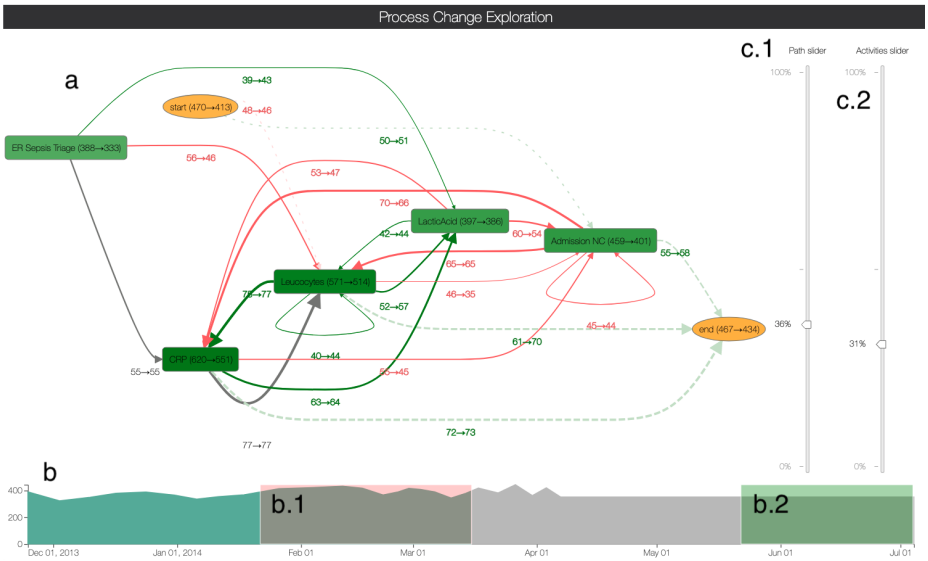


Fig. 10: Process Change Exploration (PCE) system. The interactive system consists of three parts. Part b shows the timeline of the data (green region), and forecasted values (grey region). The user can brush one or two regions (in this b.1 and b.2) on this graph to see the filtered for that time range process model or a difference between two process models. The main view (a) represents the directly follows change graph that shows the difference between two brushed regions. The c.1 and c.2 are the usual filters on the number of paths and activities to aid simplification the visual representation

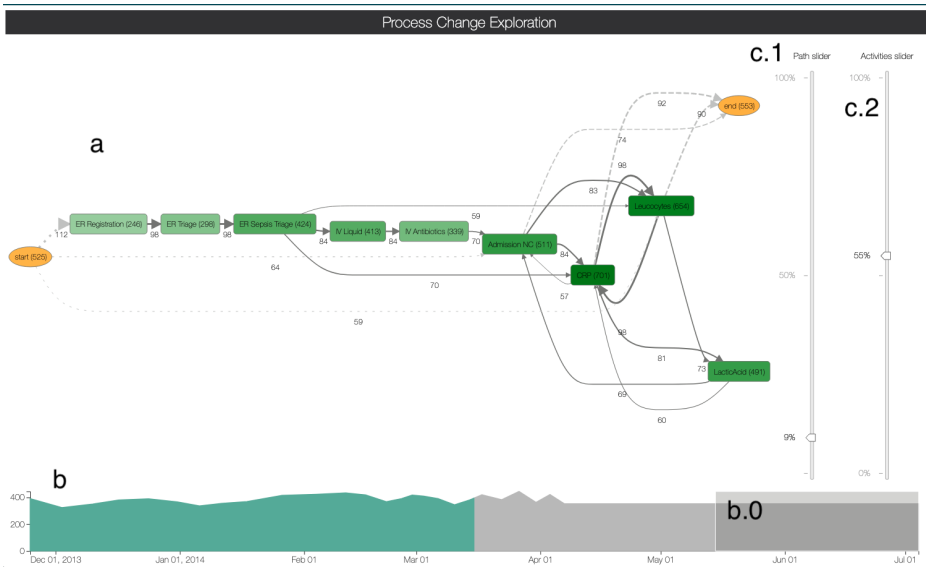


Fig. 11: Process Change Exploration (PCE) system. The interactive system consists of three parts. Part b shows the timeline of the data (green region), and forecasted values (grey region). The user can brush one time range (in this b.0) on this graph to see the filtered for that time range process model. The main view (a) represents the directly follows graph of that region. The c.1 and c.2 are the usual filters on the number of paths and activities to aid simplification the visual representation

7 Conclusion

In this paper...

References

1. van der Aalst, W.M.P., Rubin, V.A., Verbeek, H.M.W., van Dongen, B.F., Kindler, E., Günther, C.W.: Process mining: a two-step approach to balance between underfitting and overfitting. Softw. Syst. Model. 9(1), 87–111 (2010)
2. Bergmeir, C., Benítez, J.M.: On the use of cross-validation for time series predictor evaluation. Information Sciences 191, 192–213 (2012)
3. De Smedt, J., De Weerdt, J., Mori, J., Ochi, M.: Predictive process model monitoring using recurrent neural networks. arXiv preprint arXiv:2011.02819 (2020)
4. Francescomarino, C.D., Ghidini, C., Maggi, F.M., Milani, F.: Predictive process monitoring methods: Which one suits me best? In: BPM. Lecture Notes in Computer Science, vol. 11080, pp. 462–479. Springer (2018)
5. Francq, C., Zakoian, J.M.: GARCH models: structure, statistical inference and financial applications. John Wiley & Sons (2019)
6. Hanke, J.E., Reitsch, A.G., Wichern, D.W.: Business forecasting, vol. 9. Prentice Hall New Jersey (2001)
7. Hyndman, R.J., Athanasopoulos, G.: Forecasting: principles and practice. OTexts (2018)
8. Kabicher, S., Kriglstein, S., Rinderle-Ma, S.: Visual change tracking for business process models. In: ER. Lecture Notes in Computer Science, vol. 6998, pp. 504–513. Springer (2011)
9. Kriglstein, S., Rinderle-Ma, S.: Change visualizations in business processes - requirements analysis. In: GRAPP/IVAPP. pp. 584–593. SciTePress (2012)
10. Leemans, S., Poppe, E., Wynn, M.: Directly follows-based process mining: A tool. In: Proceedings of the ICPM Demo Track 2019 (CEUR Workshop Proceedings, Volume 2374). Vol. 2374. pp. 9–12. Sun SITE Central Europe (2019)
11. Leybourne, S.J., et al.: Testing for unit roots using forward and reverse dickey-fuller regressions. Oxford Bulletin of Economics and Statistics 57(4), 559–571 (1995)
12. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: Statistical and machine learning forecasting methods: Concerns and ways forward. PloS one 13(3), e0194889 (2018)
13. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: The m4 competition: 100,000 time series and 61 forecasting methods. International Journal of Forecasting 36(1), 54–74 (2020)
14. Poll, R., Polyvyanyy, A., Rosemann, M., Röglinger, M., Rupprecht, L.: Process forecasting: Towards proactive business process management. In: BPM. Lecture Notes in Computer Science, vol. 11080, pp. 496–512. Springer (2018)
15. Polyvyanyy, A., Moffat, A., García-Bañuelos, L.: An entropic relevance measure for stochastic conformance checking in process mining. In: ICPM. pp. 97–104. IEEE (2020)
16. Seabold, S., Perktold, J.: Statsmodels: Econometric and statistical modeling with python. In: Proceedings of the 9th Python in Science Conference. vol. 57, p. 61. Austin, TX (2010)
17. Shneiderman, B.: The eyes have it: A task by data type taxonomy for information visualizations. In: VL. pp. 336–343. IEEE Computer Society (1996)

18. Tax, N., Verenich, I., Rosa, M.L., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: CAiSE. Lecture Notes in Computer Science, vol. 10253, pp. 477–492. Springer (2017)
19. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: Review and benchmark. ACM Trans. Knowl. Discov. Data 13(2), 17:1–17:57 (2019)
20. Thomakos, D.D., Guerard Jr, J.B.: Naive, arima, nonparametric, transfer function and var models: A comparison of forecasting performance. International Journal of Forecasting 20(1), 53–67 (2004)
21. Weigend, A.S.: Time series prediction: forecasting the future and understanding the past. Routledge (2018)