Capturing the Sudden Concept Drift in Process Mining

Manoj Kumar M V, Likewin Thomas, and Annappa B

Department of Computer Science and Engineering
National Institute of Technology Karnataka, Surathkal
Mangalore - 575025
INDIA
{manojmv,likewinthomas}@nitk.ac.in, annappa@ieee.org
http://www.cse.nitk.ac.in

Abstract. Concept drift is the condition when the process changes during the course of execution. Current methods and analysis techniques existing in process mining are not proficient of analyzing the process which has experienced the *concept drift*. State-of-the-art process mining approaches consider the process as a static entity and assume that process remains same from beginning of its execution period to end.
Emphasis of this paper is to propose the technique for *localizing* concept drift in *control-flow* perspective by making use of *activity correlation strength* feature extracted using process log. Concept drift in the process is localized by applying statistical hypothesis testing methods. The proposed method is verified and validated on few of the real-life and artificial process logs, results obtained are promising in the direction of efficiently localizing the sudden concept drifts in process-log.
Keywords: Concept drift, process mining, event class correlation, activity correlation strength, sudden drift

1 Introduction

Process mining is a fairly new research discipline that stands between process modeling and analysis on the one hand, and computational intelligence and data mining on the other hand. The idea of process mining is to discover, monitor and improve the operational, electronic and embedded processes by using the data logged in process logs[4].

Process mining comprises (automated) process discovery (i.e., mining process models), conformance checking (i.e., monitoring deviances by matching model and log), social network/ organizational mining, automated creation of simulation models, model extension, model repair, case prediction, and history-based recommendations as shown on fig. 1.

There are two main reasons for the increasing attention in process mining. First, more and more events are being logged, thus, providing thorough info about the past of processes. Second, there is a necessity to develop and upkeep business processes in modest and quickly altering environments.

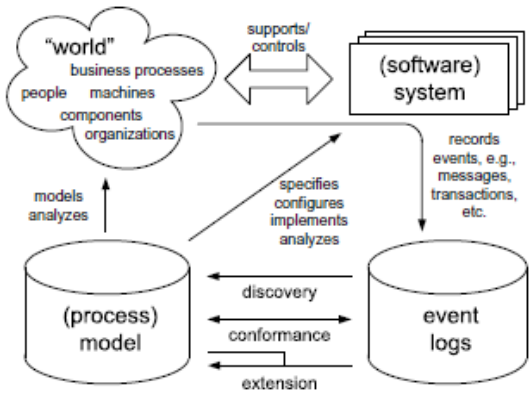


Fig. 1. Basic objectives and types of process mining [**?**]

Process mining techniques offer a means to more rigorously check compliance and ascertain the validity and reliability of information about an organization's core processes.

Beginning point for process mining is availability of appropriate event log. All process mining methods assume that it is possible to sequentially record events. Each event refers to an activity (i.e., a well-defined step in some process) and is related to a particular case (i.e., a process instance). Event logs may store extra info about events. In fact, whenever possible, process mining techniques use extra information such as the resource (i.e., person or device) executing or initiating the activity and time-stamp of the event etc.

Remaining sections of this paper are structured as follows. Section 2 discusses about concept drift with brief and concise example. Section 3 gives the brief description about the terminologies and notations used in this paper. Section 4 briefs about the methodology used to localize the sudden concept drift. Results of our experiments are given in section 5, brief about the related literature is explained in section 6 and this paper ends with some concluding remarks.

2 Concept drift

Process-centric analysis methods and techniques available in process mining are capable of generating excellent insight on working of operational process. If the process is not of static in nature, presently available process mining methods cannot be applied for the analysis. The main erroneous assumption that all of the available process mining techniques does is, *"Process at the end of its execution is same as the process at the beginning of its execution"*[12], this is not often the case due to the possibility of process change during the period of execution. All currently available process mining algorithms fail consider the changes happened in the process during the process execution.

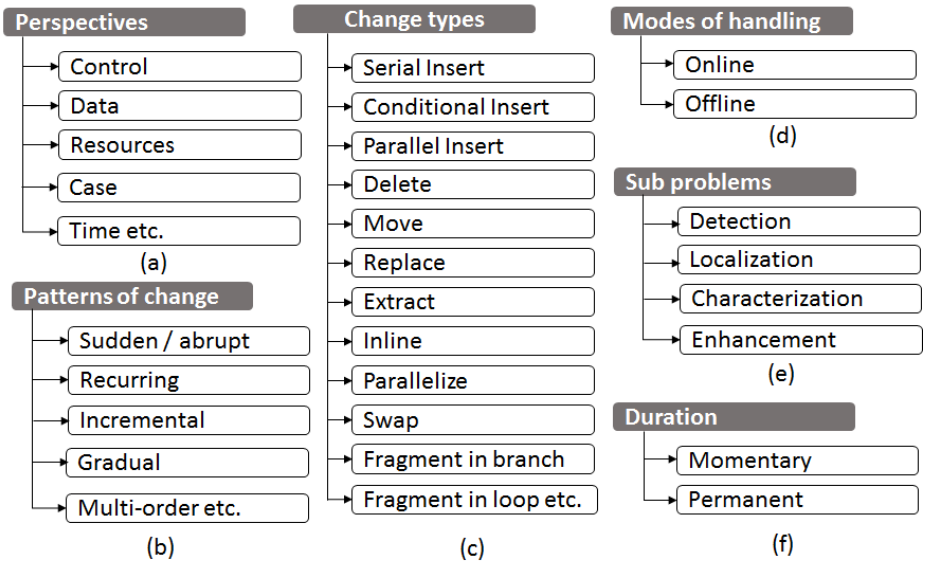


Fig. 2. Concept drift problem dimension

Possibility of occurrence of concept drift has unfortunately been neglected while proposing methods available in the area of process mining. Not concentrating and ignoring the changes in the process makes end results of analysis obsolete.

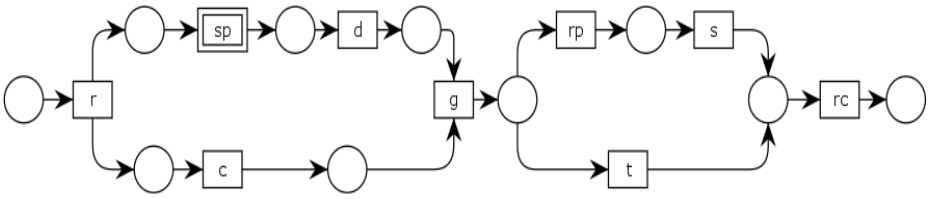
End-to-end Solution for the phenomenon of concept drift can only be achieved by considering *sub-problems involved, perspectives of change, change types, change patterns* and *duration of change* in to account, same is shown in fig. 2 *Change detection* and *change localization* are the two major *sub-problems*. *Control-flow, data, case* and *organizational* are four the main process perspectives. *Sudden, recurring, incremental* and *gradual* are the four different change types those can be normally observed. Most normally observed change *patterns of change* in control-flow perspective are shown in fig. 2(c). Please refer [7,6,5] to get to know more about different control-flow, resource and data patterns that can be observed in operational process.

For example, consider the process model shown in fig. 3(a) represent the *repair process* of electronic products in a company and is modeled with petri-net notations. A petri net is a bipartite graph consists of places (circle) and transition (rectangle). A transition becomes enable when each of its input places has at least one token in it. Upon firing of transition, it consumes a token from each of its input places and produces a token in each of its output places. The

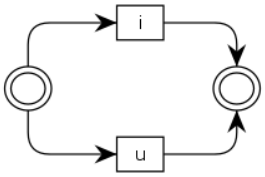
Table 1. Example process-log

Trace set-1		***Trace set-2***	
t_1	{r, i, c, d, g, rp, s, rc}	t_9	{r, i, u, c, d, g, rp , s, rc}
t_2	{r, u, d, c, g, rp, s, rc}	t_{10}	{r, u, i, c, d, g, rp, s, rc}
t_3	{r, i, c, d, g, t, rc}	t_{11}	{r, i, u, c, d, g, t, rc}
t_4	{r, u, d, c, g, t, rc}	t_{12}	{r, u, i, c, d, g, t, r, c}
t_5	{r, i, d, c, g, rp, s, rc}	t_{13}	{r, i, c, u, d, g, rp, s, rc}
t_6	{r, u, c, d, g, t, rc}	t_{14}	{r, c, i, u, d, g, t, rc}
t_7	{r, i, d, c, g, rp, s, rc}	t_{15}	{r, c, i, u, d, g, rp, s, rc}
t_8	{r, i, d, c, g, t, rc}	t_{16}	{r, i, u, d, c, g, rp, s, rc}

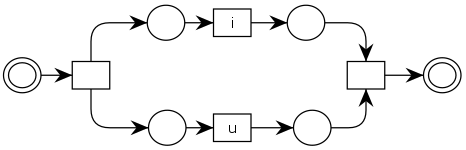
fig. shown in 3 is drawn using Colored Petri-Net[1] Tools (CPNtools[2]). Process model in fig. 3(a) has set of 10 different activities. In fig. 3(a), transition *sp* with double rectangle represents *sub-process*.



(a) Repair process modeled in petri-net process modeling notation



(b) Sub process of repair process before occurance of concept drift



(c) Sub-process of repair process after occurance of concept drift

Fig. 3. Example illustrating *sudden* concept drift in *control-flow perspective* of repair process

Activities of the process in fig. 3(a) are *r=receive repair request, i=inspect item, u=update database, c=check warranty, d=decide the cost of repair, g=get the approval from customer, rp=repair product, s=send bill and collect charges,*

[1] Coloured Petri nets (CPN) are a backward compatible extension of the concept of Petri nets. CPN preserve useful properties of Petri nets and at the same time extend initial formalism to allow the distinction between tokens.

[2] http://www.cpntools.org

t=terminate the repair process and *rc= return item and close case*. Table 1 shows the traces of the *repair process*. According to the process log shown in table 1, process experiences *concept drift* after t_8 i.e. the traces t_1 to t_8 represents the process traces before change and t_9 to t_{16} are the traces possible after process change.

Before concept drift (before t_9), any one of the activities *inspect item* or *update database* can be observed in traces of the log shown in table 1. After the occurrence of concept drift (after t_8), both *inspect item* and *update database* activities can be observed. This example precisely signify the effect of concept drift in process. If we employ the process discovery methods available in process mining to construct the process model using the process log shown in Table 1, outcome will be process model in the fig. 3 with the excerpt shown in fig. 3(a) as the subprocess replacing the activity *sp*.

3 Event class and Event class correlation

Let $\mathcal{A}$ be a set of activity names. A trace σ is a sequence of activities, i.e., $\sigma \in \mathcal{A}^*$. A simple event log L is a multi-set of traces over $\mathcal{A}$, i.e., $L \in \mathbb{B}(\mathcal{A}^*)$

Definition (Event, log trace, log). Let E be a set of unique set of log events. l is a log trace over E if and only if l is a non-repeating sequence on E . A set of log traces L is a log over E if and only if all log traces $l \in L$ are log traces over E and $\forall_{l_1,l_2} \in L : (set(l_1) \cap set(l_2) \neq \emptyset) \rightarrow (l_1 = l_2)$.

Using the definition of event, trace and log, event class can be defined as follows.

Definition (Event class). $c \in E \rightarrow C$ maps each event to its event class, where C is the set of event classes.

The set of event classes for a log trace l can be defined as follows:

$$C(l) = \{c(e)|e \in l\}$$

The set of event classes for a log L is defined as follows:

$$C(L) \cup_{l \in L} C(l)$$

Let C be a set of event classes. The function $ecc \in C \times C \rightarrow \mathbb{R}_0^+$ assigns to each tuple of event classes a certain correlation value. The larger this the value is, the more related the two respective event classes are.

In our method we define the correlation function among event classes by scanning the whole log. We begin with a matrix of $C \times C$, set with zero values before the real scanning pass. While traversing the log, this matrix is updated for every following relation that is found. Correlation matrix, as well as the correlation function itself, is symmetric, i.e., $ecc(X, Y) = ecc(Y, X)$. During the scanning pass, this regularity requires to be preserved by the algorithm.

Consider the fig. 4, the scanning is presently examining an event of class e_1. We call the event presently under consideration as *reference event*. Looking at the directly preceding event of class e_2, the scanner can establish an observation

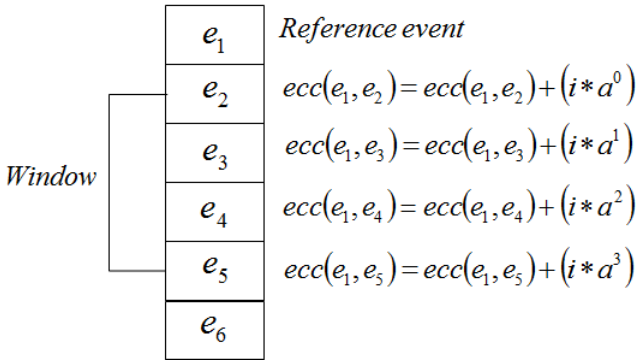


Fig. 4. *ecc* calculation

of the co-occurrence between event classes e_1 and e_2, which means that their association is strengthened. Similarly, the correlation matrix value for $ecc(e_1, e_2)$ is incremented by i, the increment value (generally set to 1). In our method, the scanning pass uses a look-forward window for calculating each event. This means that if the look-forward windows size is seven, the scanner will consider the upcoming seven events which have followed the reference event. When calculating events in the look forward window, the scanner will weaken its measurement exponentially, based on an attenuation factor a, where $0 \leq a \leq 1$.

For any event y in the look-forward window, where x is the reference event, the correlation matrix will be updated as given below

$$ecc(c(x), c(y)) = ecc(c(x), c(y)) + (i.a^n) \tag{1}$$

where n is the number of events located between x and y in the trace.

After the scanning pass has estimated all events in all traces of the log, a trustworthy correlation function between event classes is recognized, as expressed in the aggregated correlation matrix. Our correlation function thus relates two event classes as more linked, if events of these classes commonly happen closely together in traces of the log.

Concept drift is the condition where the process experiences change during the course of analysis. We believe that the representative appearance of feature values change before and after the occurrence of concept drift. By considering the sequential order of process instances in the log, we apply *windowing* strategy for selecting the instances for processing and to localize the occurrence of concept drift. *Statistical hypothesis tests*[3] are used to examine differences between successive feature values obtained using *event class correlation.*

4 Methodology

[3] Hypothesis testing is really a systematic way to test claims or ideas about a group or population, using data measured in a sample.

Algorithm 1 Algorithm to detect concept drift using event class correlation

Require: Process log with concept drifts

1: $sub\_logs \leftarrow 0$ // set the initial value to 0
2: $sub\_logs \leftarrow split\_log(process\_log, size)$
3: $num\_sub\_logs \leftarrow sub\_logs.size()$
4: **while** $num\_sub\_logs \neq 0$ **do**
5: $i \leftarrow 0$
6: $activities = get\_activities\_of\_sub\_log(sub\_log[i])$ // get the number of activities in the each sub log
7: $i \leftarrow i + 1$
8: $cor[size(activities)][size(activities)] \leftarrow 0$
9: **for** $\forall case_i \in sub\_logs_i$ **do**
10: $sub_case \leftarrow 0$
11: **for** $\forall event_i \in case_i$ **do**
12: $look\_back \leftarrow 0$ l
13: **for** $\forall events_j \in case_i$ **do**
14: **if** name$(event_i \neq event_j)$ **then**
15: **if** $look\_back \leq size$ **then**
16: $cor[event_i][event_j] \leftarrow cor[event_i][event_j]+(i \times a^{look\_back})$ // calculate the ecc of event classes i and j
17: $look\_back = look\_back + 1$
18: **end if**
19: **end if**
20: **end for**
21: **end for**
22: **end for**
23: $num\_sub\_logs \leftarrow num\_sub\_logs - 1$
24: $level\_of\_significance = 0.05$ // Set the level of significance (alpha value)
25: $Test\_satistic = test\_hypothesis(cor, hypothesis\_test\_name, window\_size, num\_of\_popultions)$ // (performing hypothesis tests)
26: $P\_value \leftarrow Compute\_P - value(Test\_statistic)$
27: **if** $P\_value \leq level\_of\_significance$ **then**
28: Reject $\mathcal{H}_0$ and declare concept drift // deciding the validity of $\mathcal{H}_0$
29: **end if**
30: **end while**

The standard process of statistical hypothesis testing comprises of four phases

- $\boldsymbol{S}_1$**:** Formulating *null* ($\mathcal{H}_0$) and *alternative* hypothesis ($\mathcal{H}_1$)
- $\boldsymbol{S}_2$ **:** Identifying a *test statistic* that can be used to assess the trustworthiness $\mathcal{H}_0$.
- $\boldsymbol{S}_3$ **:** Calculate the P-value (probability of obtaining a sample outcome, given that the $\mathcal{H}_0$ is true).
- $\boldsymbol{S}_4$ **:** Compare the P-value to a statistical significance level α. If $P \leq \alpha$, that the observed effect is statistically significant, $\mathcal{H}_0$ is ignored, and the $\mathcal{H}_1$ is considered as valid.

$\mathcal{H}_0$ can be stated as,

($\mathcal{H}_0$): There is no significant characteristic differences in the manifestation of consecutive populations of *feature* values.

Null hypothesis is considered as fact until proved as false. When the null hypothesis is proved as false, alternative hypothesis ($\mathcal{H}_1$: There is significant difference in manifestation of feature values) is considered and accepted and occurrence concept drift is declared.

Complete procedure for assessing the hypothesis tests on consecutive populations of *ecc* values is shown in the algorithm 1. We choose *two-sample* (since we need to analyze two samples of the population at the given point of time for detecting concept drift), *independent* (since both the samples are not depending on each other), *non-parametric*(since we do not know the priori distribution of the feature values in an event log), *uni-variate* and *multi-variate* (univariate tests deal with scalar data and multivariate tests deal with vector data) statistical hypothesis tests for detecting and localizing the concept drift in the process.

Using windowing strategy as instance selection method, successive populations of feature values are compared and examined to discover any significant difference. Significant difference between feature values only observed during the change in the process. Depending on the requirement of our problem and based on the characteristics of the tests described in the previous paragraph we consider *Mann-Whitney U Test* and *The Moses Test for Equal Variability. Mann-Whitney U Test* is used to answer ***"do two independent samples represent two populations with different median values"*** (or different distributions with respect to the rank-orderings of the scores in the two underlying population distributions)? *The Moses Test for Equal Variability* test will be used to answer ***Do two independent samples represent two populations with different variances?***

5 Experiments and Results

Table 2. Process-log details and concept drift locations

	Process log	**Cases**	**Activities**	**Events**	$\mathbf{c}_{reo}$	$\mathbf{c}_{rep}$	$\mathbf{c}_{ins}$	$\mathbf{c}_{del}$
L_1	Loan application process	13,087	36	2,62,200	5,000	7,500	-	-
L_2	Volvo IT incident management process	7,554	13	65,533	-	3,000	4,000	-
L_3	Insurance claim process	500	21	7,033	-	-	200	400

Process before the occurrence of concept drift represent different version of the process than after the occurrence of concept drift. Concept drift can be observed in the process any number of times.

It is very hard to find real-life operational process-log with concept drift in it. Process mining doesn't has any standard data set or workbench for testing the credibility of algorithms detecting and localizing concept drift. There are

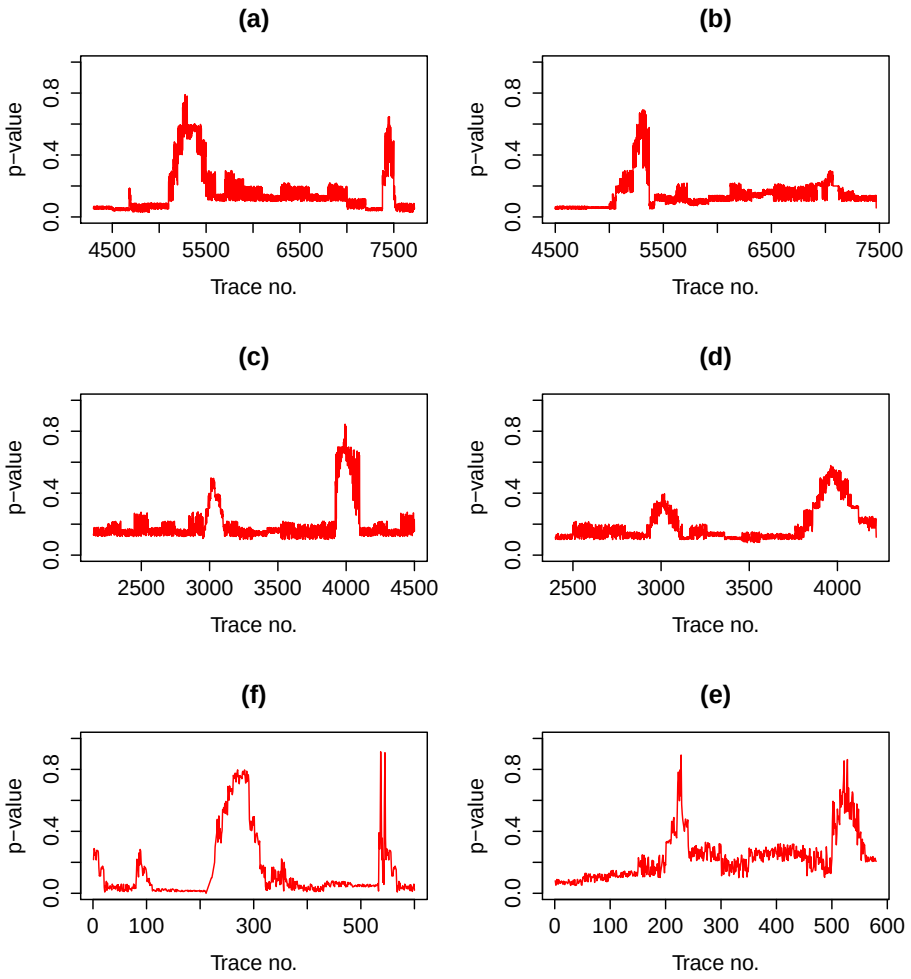


Fig. 5. Result of application of statistical hypothesis testing on process logs L_1, L_2 and L_3 given in 2. Graphs (a),(c),(e) are the results of application of *Mann-Whitney U Test* and the graphs (b), (d), (f) are the results of application of *The Moses Test for Equal Variability* on process logs L_1, L_2 and L_3 respectively

few real-life standard datasets available[3] [4], but they are not appropriate for testing the algorithms dealing with concept drift. In our experiments, we have taken appropriate data sets from open repository of process logs and artificially induced concept drift in the control-flow perspective of the process.

[3] http://data.3tu.nl/repository/
[4] http://www.processmining.org/logs/start

Process logs form the open process log repository are used and modified to include concept drift. We used Colored Petri Net (CPN) Tools with CPNXES library[5] for creating synthetic process logs. Approach proposed in this paper is tested on 3 different logs shown in table 2.

- $\boldsymbol{c_{reo}}$**:** Rearranging activities.
- $\boldsymbol{c_{rep}}$**:** Replacing one activity with other.
- $\boldsymbol{c_{ins}}$**:** Inserting a new activity.
- $\boldsymbol{c_{del}}$**:** Deleting an existing activity.

Table 2 lists info about three different process logs that we have considered to validate the approaches proposed in this paper. Process logs L_1 (*Loan application process*) and L_2(*Volvo IT incident management process*) are taken from on-line process log repository. Process log L_3 (*Insurance claim process*) is generated with the help of CPNTools and CPNXES library. Both L_1 and L_2 are modified to include concept drift[1]. Table 2 shows the details about location of concept drift in L_1, L_2 and L_3.

Process log L_1 is altered to include the concept drift caused by sudden c_{reo} and c_{rep}. L_2 is altered to include sudden c_{rep} and c_{ins}. Process log L_3 is altered to include sudden c_{ins} and c_{del}. By considering the limitation of space, we have ignored to include the models of the process L_1, L_2 and L_3 in this paper.

$\mathcal{H}_0$ is assessed on each of L_1, L_2 and L_3 by applying hypothesis tests discussed in the last section, result is shown in fig. 5. Graphs a,c,f in fig. 5 are the results of application of *Mann-Whitney U Test* and graphs b,d,e in fig. 5 are the results of application of *The Moses Test for Equal Variability* on process logs described in the table 2. Crests in the graphs given in fig. 5 signify real concept drift in the process. There are some false alarms are also be possible, one should be very careful about the actual concept drift and the concept drift caused by noise in the process log. The drastic change in the P-value is observed during the occurrence of concept drift. Our technique perform poorly when applied to localize the concept drift caused by c_{rep} and same can be seen in the graphs shown in fig. 5 (b) and (c) (at the trace number 7500 in 5 (b) and 3000 in 5 (c)). Overall, results shown in graph 5 implies that the methods presented in this paper for localizing concept drift is promising. In future, the proposed technique is going to be added to ProM, and there by making it possible for people to experiment with it.

6 Related work

The word *concept drift* is initially coined by Schlimmer et.al. during 1986 in the article *Incremental learning from noisy data*[8]. Phenomenon of concept drift is known by many terminologies in other research disciplines (as Covariate Shift

[5] https://westergaard.eu/2011/07/prom-package-documentation-keyvalue/

[1] process logs and models used in this paper can be downloaded at http://www.cse.nitk.ac.in/researchscholars/manoj-kumar-m-v

in machine learning, as Load Shedding in databases, as Temporal Evolution in Information retrial etc.). Efficiently handling concept drift is an important concern in every data analysis disciplines[2], unfortunately it has been deeply neglected in process mining. According to [2], concept drift is a *non stationary learning problem over time* and [1] describes drift as the *process of changing the process*. The core theory when dealing with the concept drift problem is *uncertainty* about the future. It can be *assumed, estimated or predicted but there is no certainty*.

Some efforts have been made to find different versions of control-flow perspective of the process using clustering and classification techniques available in Data Mining[9,10,11]. Finding different versions of the process does not consider the *type, pattern* and *perspective* of concept drift. Hence, they cannot be the suitable means for solving phenomenon of concept drift.

ProM is the open source process mining framework consisiting more than $1,200$[6] plug-in and plug-in variants that can be used for solving different process mining problems, out of which one or two plug-ins capable of addressing the problem of concept drift. To our knowledge, two works in the literature that addresses concept drift in process mining are [12,14,13]. Technique proposed in [12] are tested on real setting and the results are documented in [13]. Both [12,13] proposes extracting different global and local features out of process log and applying statistical hypothesis testing for detecting and localizing concept drift. Techniques shown in [12,14] propose solution for *offline* and *online* methods for detecting and localizing *sudden* concept drift in *control-flow* perspective of process. The idea of extracting Event Class Correlation (ecc) feature is taken form [3]. End-to-end solution for the problem of concept drift can only be accomplished if it is addressed by considering all *perspectives, types and patterns* of change shown in fig.2. Effort given in this paper suggests the method of localizing sudden concept drift in the control-flow perspective of the process using event class correlation feature by applying statistical hypothesis testing methods.

7 Conclusion

Handling the phenomenon of concept drift efficiently is the prime concern in all disciplines that deal with data analysis. Concept drift is the situation when process experiences changes in its associated perspectives during the period of its execution. The configuration of the process before the occurrence of concept drift is different from the process after the occurrence of concept drift. State-of-the-art process-centric analysis techniques available in process mining behave poorly when employed to analyze the process that has experienced concept drift. Because, they consider the process as a static entity. But, process represents the dynamic aspect of the organization and can evolve in any perspective showing any change pattern exhibiting several different change type during the phase of its execution. This paper proposes the extraction of *event class correlation*

[6] http://www.promtools.org/doku.php?id=packdocs

feature for localizing the sudden concept drift in the control-flow perspective of the operational process. Results of the experimental study shown that proposed methods are capable of localizing concept drift efficiently. Our feature work include extension of the proposed methods to make working in on-line setting for sudden and gradual drift detection and localization.

References

1. Gama, Joo, et al. *"A survey on concept drift adaptation."* ACM Computing Surveys (CSUR) 46.4 (2014): 44.
2. Zliobaite, Indre. *Learning under concept drift: an overview. Overview*, Technical report, Vilnius University, 2009 techniques, related areas, applications Subjects: Artificial Intelligence, 2009.
3. Gnther, Christian W., Anne Rozinat, and Wil MP Van Der Aalst. *"Activity mining by global trace segmentation."* Business process management workshops. Springer Berlin Heidelberg, 2010.
4. Van Der Aalst, Wil, et al. "Process mining manifesto." Business process management workshops. Springer Berlin Heidelberg, 2012.
5. Russell, Nick, Ter Hofstede, Arthur HM, Mulyar, Nataliya, *Workflow controlflow patterns: A revised view*, Citeseer, 2006.
6. Ter Hofstede, Arthur HM, David Edmond, and Wil MP van der Aalst. *"Workflow resource patterns"* (2005): 13-17.
7. Russell, Nick, Arthur HM Ter Hofstede, David Edmond, and Wil MP van der Aalst. *Workflow data patterns*. QUT Technical report, FIT-TR-2004-01, Queensland University of Technology, Brisbane, 2004.
8. Schlimmer, Jeffrey C., and Richard H. Granger Jr. *"Incremental learning from noisy data."* Machine learning 1.3 (1986): 317-354.
9. Song, Minseok, Christian W. Gnther, and Wil MP Van der Aalst. *"Trace clustering in process mining."* Business Process Management Workshops. Springer Berlin Heidelberg, 2009.
10. Luengo, Daniela, and Marcos Seplveda. *"Applying clustering in process mining to find different versions of a business process that changes over time."* Business Process Management Workshops. Springer Berlin Heidelberg, 2012.
11. Bose, RP Jagadeesh Chandra, and Wil MP van der Aalst. *"Context Aware Trace Clustering: Towards Improving Process Mining Results."* SDM. 2009.
12. Bose, RP Jagadeesh Chandra, et al. *"Handling concept drift in process mining."* Advanced Information Systems Engineering. Springer Berlin Heidelberg, 2011.
13. Bose, RP Jagadeesh Chandra, et al. *"Dealing with concept drifts in process mining."* Neural Networks and Learning Systems, IEEE Transactions on 25.1 (2014): 154-171.
14. Carmona, Josep, and Ricard Gavalda. *"Online techniques for dealing with concept drift in process mining."* Advances in Intelligent Data Analysis XI. Springer Berlin Heidelberg, 2012. 90-102.