EVALUATION OF PROCESS MODELS USING HEURISTIC MINER AND DISJUNCTIVE WORKFLOW SCHEMA ALGORITHM FOR DYEING PROCESS

Saravanan .M.S[1] and Rama Sree .R.J[2]

[1]Research Scholar in R & D Centre, Bharathiar University, Coimbatore, Tamil Nadu, INDIA.
Asst. Prof. in Dept. of I.T in Vel Tech Dr. RR & Dr. SR Technical University, Chennai, INDIA.
saranenadu@yahoo.co.in

[2]Professor & Head in the Department of Computer Science, Rashtriya Sanskrit Vidyapeetha, Tirupati, Andhra Pradesh, INDIA.
rjramasree@yahoo.com

ABSTRACT

The dyeing domain is a dynamic, complex and involved various discipline. The dyeing process is difficult to automate processes, because of their interdisciplinary and dynamic in nature. Moreover, it is also critical to keep a check on the automated processes to produce the expected results in the form of quality and timely dyeing processes. Delivering these processes is a complex, because colour shades are generally new, and tend to have complex dye mix problems. The drive in dyeing organizations is to reduce costs and at the same time improve the quality of colours the shades need. The benefit of workflow management systems in such domain includes cost reduction, improved operational efficiencies, pH test error reduction, improved shade quality, better communication and collaboration and real time audit of processes. Hence, such systems need a process model and this process model should clearly depict the control flow of the tasks in any type of dyeing process. Therefore the focus of this paper is on the process discovery. For this a plug-in implemented in ProM Framework has been used, for this HM algorithm and DWS algorithm. This study also evaluates the HM and DWS mining algorithms, which may have a major impact on the future development of efficient process mining algorithms.

KEYWORDS

Dynamic, complex, colour, shades, dye mix, real time audit, HM and DWS algorithm.

1. INTRODUCTION

A good managerial approach considers business processes to be strategic assets of an organization that must be understood, managed and improved to deliver value added products and services to clients or customers. This is very similar to “quality management” or “continues improvement process” approaches. These business processes are able to integrate a “change capability” to an organization with human and technological viewpoints. The organization goal is time dependent, so the business processes need to improve or require changes with respect to time. Therefore these processes need to identify and verify the new opportunities and changes are required to gain more revenue or avoid loss of every organization.

DOI : 10.5121/ijitcs.2011.1304

Dyeing unit has various type of processing method to make colour for the cotton yarn with respect to the shade and quality. The most used method is cabinet dyeing processing system. The cabinet dyeing processing is not as simple as it sounds. In this section, let us take a look at the characteristics of the dyeing unit domain.

1.1 Dynamic, complex and cross-functional processes

Dyeing unit processes involve various colouring treatment and administrative processes, large volumes of data and a large number of people, customers and personnel. There are also financial tasks linked with these processes. It is apparent that dyeing unit processes are just not only related to the coloring the cotton yarn, but they also involve procedures from other disciplines like management, finance, IT etc. Moreover a treatment process may be dynamic and can become complex. For instance, a coloring process applied for a particular shade X but during coloring treatment it may develop some other color like Y, therefore the process of treatment cannot be viewed as a simple sequential process. It may consist of various colouring treatment conditions concurrently and may also involve personnel from various disciplines.

1.2. Issues concerning automation, collaboration and coordination

Most of the activities in each process can only be partially automated as many trade-offs, decisions and actions must be performed by dyeing experts called dyers and cannot be automated or even partially delegated to automated means. Besides automation issues, the quality, degree of collaboration and coordination among workers and between workers [1].

1.3. Improper data management

The management of the dyeing unit and dyeing experts called dyers both suffer from data overload. The data is often redundant, inaccurate, uninformative or confusing. Thus, it is difficult to keep up with the increasing demand for reliable, broadly available dyeing process information.

1.4. Simultaneous functioning of various applications

The dyeing units has many applications that support specific functions and these applications often reside within them and is not easily made available to other processes that may require similar data. This also increases data redundancy and also makes the situation confusing because different applications may store same data about the same entity, e.g. a color dye mix details can be stored in the various places such as pre treatment, treatment and post treatment etc.

1.5. Ad hoc actions and process changes

In a dyeing unit the actions are influenced by organizational and dyeing process pollution events, introduction of new administrative procedures, dyeing method approaches and technological developments. These events force users like dyers, supervisors, and other staff members to change, extend or discontinue the usual procedures. Such unstructured and ad hoc actions are difficult to model and automate. In such cases, it becomes critical and important that to taken care of the changes in the dyeing unit processes [1]. Hence, the dyeing processes in the dyeing domain are dynamic, ad-hoc, unstructured and multi-disciplinary in nature. Also in modern times, dyeing organizations place strong emphasis on reducing cost, pollution for organizational efficiency and effectiveness to control their dyeing expenditures. They aim at providing world class services at low cost. In such a situation, it becomes of utmost importance to evaluate the existing infrastructure of the services being offered by these organizations. This is where process mining techniques can be employed to extract process models from the event logs [2] of information systems deployed in dyeing organizations.

2. RELATED WORK

The idea of process mining is not new [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. Cook and Wolf have investigated similar issues in the context of software engineering processes. In [3] they describe three methods for process discovery: one using neural networks, one using a purely algorithmic approach, and one Markovian approach. The authors consider the latter two the most promising approaches. The purely algorithmic approach builds a finite state machine where states are fused if their futures that are in terms of possible behavior in the next k steps are identical. The Markovian approach uses a mixture of algorithmic and statistical methods and is able to deal with noise. Note that the results presented in [3] are limited to sequential behavior. Related, but in a different domain, is the work presented in [15], [16] also using a Markovian approach restricted to sequential processes. Cook and Wolf extend their work to concurrent processes in [5]. They propose specific metrics such as entropy, event type counts, periodicity, and causality and use these metrics to discover models out of event streams. However, they do not provide an approach to generate explicit process models. In [6] Cook and Wolf provide a measure to quantify discrepancies between a process model and the actual behavior as registered using event based data

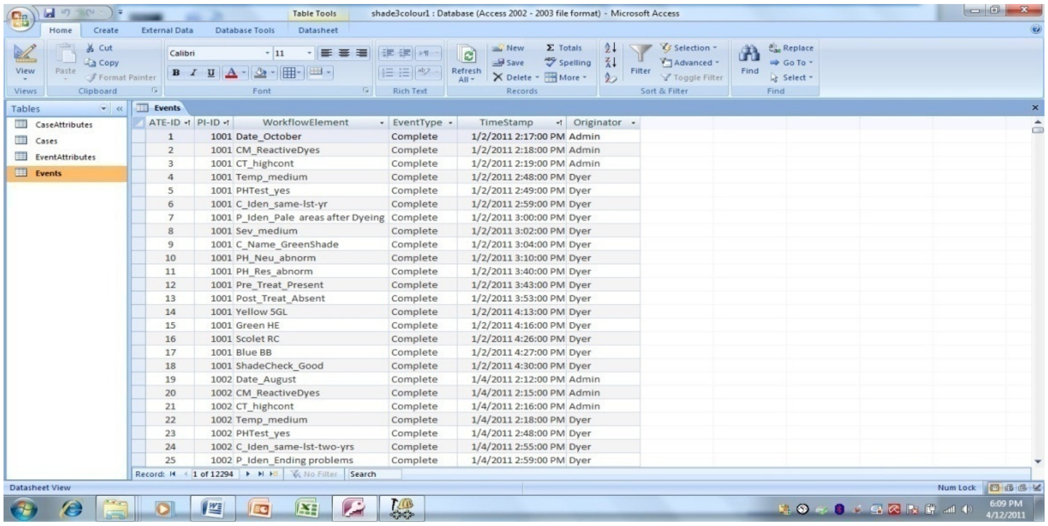
It is also critical to keep a check on the automated processes to produce the expected results in form of quality and timely dyeing processes. Delivering these processes is a complex because shades are generally new and tend to have complex dye mix problems [17]. The drive in dyeing organizations is to reduce costs and at the same time improve the quality of shades. The benefit of workflow management systems in such domain includes cost reduction, improved operational efficiencies, clinical error reduction, improved shade quality, better communication and collaboration and real time audit of processes [17]. Therefore, the process aware systems such as workflow management systems need a process model and this process model should clearly depict the control flow of the tasks in any business process. Therefore, focus of this paper is process discovery of dyeing process and evaluation of these workflow using HM and DWS mining algorithms. For this a plug-in implemented in ProM has been used, that is HM plug-in and DWS plug-in.

So these HM and DWS algorithms first implemented for Emerald dyeing unit dyeing processes and it has lot of noise due to human errors, incompleteness of data etc. Hence, HM was chosen to investigate the processes in Emerald dyeing because it was one of the robust algorithms to till date. The focus of experimentation with this algorithm is to obtain a process model for the investigated dyeing process as well as to evaluate the performance of this plug-in for the input shade data. The structure of this paper is as follows. Section 3 introduces Dyeing unit workflow analysis. Section 4 elaborates the fundamentals of the HM algorithm, explains its implementation in the form of a mining plug-in and experiments with HM algorithm. Section 5 describes the DWS algorithm, implementation and experiment with DWS algorithm. The choice for the DWS algorithm was made because it provides process models specific to a group of similar shades present in the log. This would help us to explore smaller and specific colouring process models. The focus of experimentation with this algorithm is to obtain a process model for the investigated dyeing process as well as to evaluate the performance of this plug-in for the shade data. Section 6 concludes the paper by summarizing the findings.

3. DYEING UNIT WORKFLOW ANALYSIS

The Dyeing unit workflow is analyzed of dyeing process in the context of process mining. The Dyeing experts called Dyers are doing the treatment process to produce different shades of the colours. The Dyeing units are processing the colour to cotton yarn. The dyeing methods are categorized in to three types (i) Manual Dyeing (ii) Cone Dyeing and (iii) Cabinet Dyeing. The manual and cabinet dyeing method is mostly used by the small scale industries and dyeing units. The cabinet dyeing units are broadly classified into (i) Simple (ii) Moderate and (iii) Complex, based on the complexities, functionality, turnover and infrastructure, since the proposed research study is first of its kind moderate cabinet dyeing unit have been considered. The dyeing units namely Emerald and Jayabala are consulted for acquiring the real time input and output process models. The Emerald and Jayabala dyeing units are located at Nagari and Sathrawada respectively in Andhra Pradesh of India. The data collected from Emerald and Jayabala dyeing units is classified at various levels to learn the feature values using Heuristic Miner and Disjunctive Workflow Schema algorithms in Process Mining Framework. HM and DWS mining algorithms take event logs as input and produce output process models.

This data is received in form of the database in Microsoft Access. It contains various tables recording information about shades, pretreatment, colour mixers, post treatment, various pH tests etc. The following figure shows a list of tables contained in this database.



ATE-ID	PI-ID	WorkflowElement	EventType	TimeStamp	Originator
1	1001	Date_October	Complete	1/2/2011 2:17:00 PM	Admin
2	1001	CM_ReactiveDyes	Complete	1/2/2011 2:18:00 PM	Admin
3	1001	CT_highcont	Complete	1/2/2011 2:19:00 PM	Admin
4	1001	Temp_medium	Complete	1/2/2011 2:48:00 PM	Dyer
5	1001	PHTest_yes	Complete	1/2/2011 2:49:00 PM	Dyer
6	1001	C_Iden_same-lst-yr	Complete	1/2/2011 2:59:00 PM	Dyer
7	1001	P_Iden_Pale areas after Dyeing	Complete	1/2/2011 3:00:00 PM	Dyer
8	1001	Sev_medium	Complete	1/2/2011 3:02:00 PM	Dyer
9	1001	C_Name_GreenShade	Complete	1/2/2011 3:04:00 PM	Dyer
10	1001	PH_Neu_abnorm	Complete	1/2/2011 3:10:00 PM	Dyer
11	1001	PH_Res_abnorm	Complete	1/2/2011 3:40:00 PM	Dyer
12	1001	Pre_Treat_Present	Complete	1/2/2011 3:43:00 PM	Dyer
13	1001	Post_Treat_Absent	Complete	1/2/2011 3:53:00 PM	Dyer
14	1001	Yellow 5GL	Complete	1/2/2011 4:13:00 PM	Dyer
15	1001	Green HE	Complete	1/2/2011 4:16:00 PM	Dyer
16	1001	Scolet RC	Complete	1/2/2011 4:26:00 PM	Dyer
17	1001	Blue BB	Complete	1/2/2011 4:27:00 PM	Dyer
18	1001	ShadeCheck_Good	Complete	1/2/2011 4:30:00 PM	Dyer
19	1002	Date_August	Complete	1/4/2011 2:12:00 PM	Admin
20	1002	CM_ReactiveDyes	Complete	1/4/2011 2:15:00 PM	Admin
21	1002	CT_highcont	Complete	1/4/2011 2:16:00 PM	Admin
22	1002	Temp_medium	Complete	1/4/2011 2:18:00 PM	Dyer
23	1002	PHTest_yes	Complete	1/4/2011 2:48:00 PM	Dyer
24	1002	C_Iden_same-lst-two-yrs	Complete	1/4/2011 2:55:00 PM	Dyer
25	1002	P_Iden_Ending problems	Complete	1/4/2011 2:59:00 PM	Dyer

Record: 1 of 12294 No Filter Search

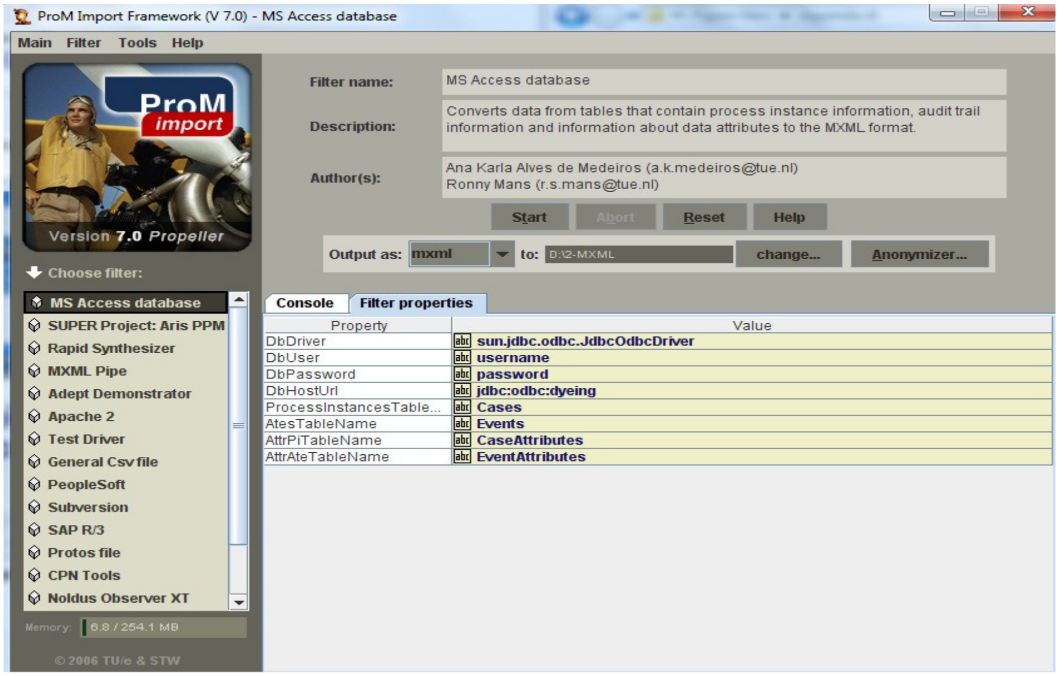
Figure 1. Tables in the Emerald Dyeing Unit Database

Besides taking an overview of the table, some of the observations about the data contained in the database are summarized below.

In the database, data is categorized under numerous headings (fields) but not all of these contain data. For example, there are numerous combinations that any shade can have from but the number of shades have from these many combinations is low. In process mining terms it means that the number of events per case is very less than the total number of events in the log. This shows that a lot of highly low-frequent events are present in the event log.

Due to a high degree of low-frequent behavior, it is uncertain because it may be due to some human error such that erroneous insertions or non-insertions of events or it actually low frequent. The possibilities of noise in real world databases like Emerald dyeing unit processes cannot be ignored but this possible presence of noise cannot be distinguished from actual process characteristics.

After having an overview of the data in Emerald dyeing unit dyeing process, it is understood that these tables cannot be directly used for experimentation in ProM, because it accepts data only in MXML format. The conversion of MS-Access data to MXML is achieved by the MS-Access import plug-in as shown in Figure 2 implemented in the ProM Import framework. Detailed explanation about this import plug-in is not the focus of this paper, so the readers are referred to [18] to read more about this conversion process. The converted MS-Access table from Emerald dyeing unit processes and its corresponding MXML log is shown in Figure 3. The logs used for various experiments pertain to the different events that occur for numerous shades. For example, some logs represent the route followed by combinations for different shades, and some logs may pertain to various color mix or treatment activities etc.



Filter name: MS Access database

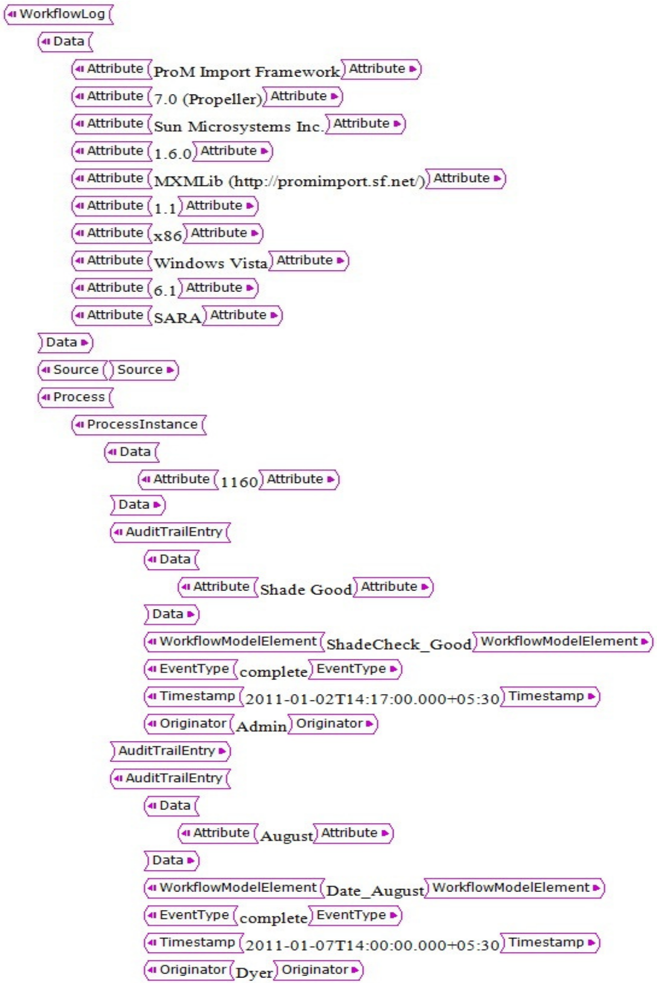
Description: Converts data from tables that contain process instance information, audit trail information and information about data attributes to the MXML format.

Author(s): Ana Karla Alves de Medeiros (a.k.medeiros@tue.nl)
Ronny Mans (r.s.mans@tue.nl)

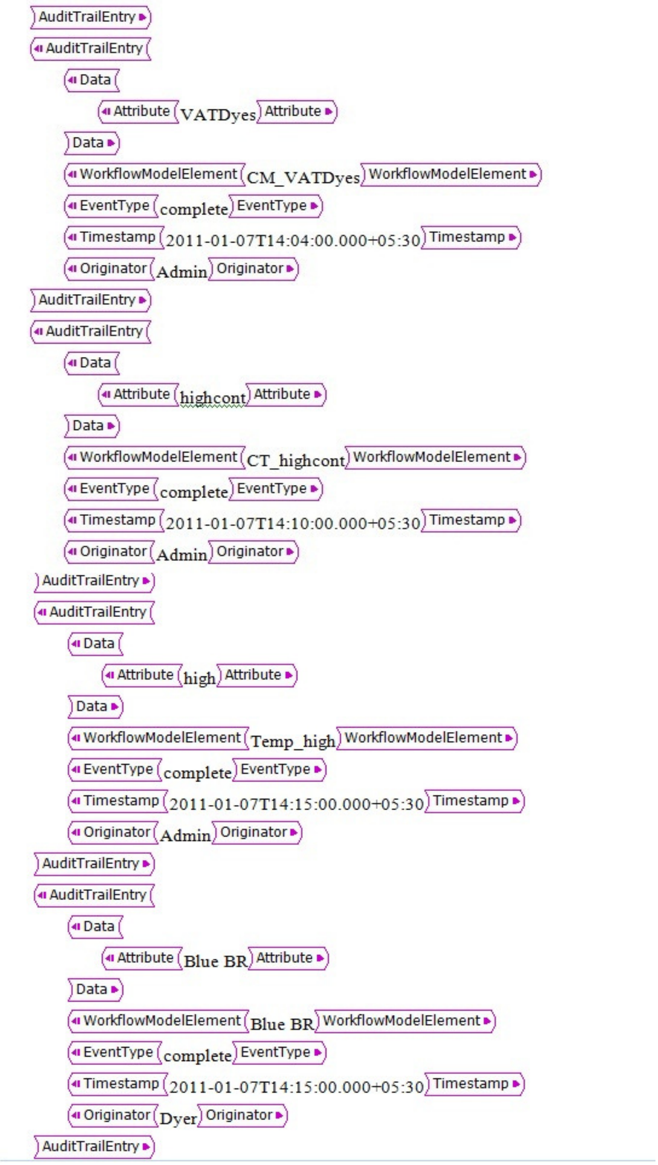
Output as: mxml to: D:\2-MXML

Property	Value
DbDriver	sun.jdbc.odbc.JdbcOdbcDriver
DbUser	username
DbPassword	password
DbHostUrl	jdbc:odbc:dyeing
ProcessInstancesTable...	Cases
AtesTableName	Events
AttrPiTableName	CaseAttributes
AttrAteTableName	EventAttributes

Figure 2. ProM Import Screen Shot to convert MS-Access Table in to MXML format



```
WorkflowLog
    Data
        Attribute ProM Import Framework Attribute
        Attribute 7.0 (Propeller) Attribute
        Attribute Sun Microsystems Inc. Attribute
        Attribute 1.6.0 Attribute
        Attribute MXMLib (http://promimport.sf.net/) Attribute
        Attribute 1.1 Attribute
        Attribute x86 Attribute
        Attribute Windows Vista Attribute
        Attribute 6.1 Attribute
        Attribute SARA Attribute
    Data
    Source  Source
    Process
        ProcessInstance
            Data
                Attribute 1160 Attribute
            Data
            AuditTrailEntry
                Data
                    Attribute Shade Good Attribute
                Data
                WorkflowModelElement ShadeCheck_Good WorkflowModelElement
                EventType complete EventType
                Timestamp 2011-01-02T14:17:00.000+05:30 Timestamp
                Originator Admin Originator
            AuditTrailEntry
            AuditTrailEntry
                Data
                    Attribute August Attribute
                Data
                WorkflowModelElement Date_August WorkflowModelElement
                EventType complete EventType
                Timestamp 2011-01-07T14:00:00.000+05:30 Timestamp
                Originator Dyer Originator
```



```
AuditTrailEntry ▸
◂ AuditTrailEntry
    ◂ Data
        ◂ Attribute VATDyes Attribute ▸
    Data ▸
    ◂ WorkflowModelElement CM_VATDyes WorkflowModelElement ▸
    ◂ EventType complete EventType ▸
    ◂ Timestamp 2011-01-07T14:04:00.000+05:30 Timestamp ▸
    ◂ Originator Admin Originator ▸
AuditTrailEntry ▸
◂ AuditTrailEntry
    ◂ Data
        ◂ Attribute highcont Attribute ▸
    Data ▸
    ◂ WorkflowModelElement CT_highcont WorkflowModelElement ▸
    ◂ EventType complete EventType ▸
    ◂ Timestamp 2011-01-07T14:10:00.000+05:30 Timestamp ▸
    ◂ Originator Admin Originator ▸
AuditTrailEntry ▸
◂ AuditTrailEntry
    ◂ Data
        ◂ Attribute high Attribute ▸
    Data ▸
    ◂ WorkflowModelElement Temp_high WorkflowModelElement ▸
    ◂ EventType complete EventType ▸
    ◂ Timestamp 2011-01-07T14:15:00.000+05:30 Timestamp ▸
    ◂ Originator Admin Originator ▸
AuditTrailEntry ▸
◂ AuditTrailEntry
    ◂ Data
        ◂ Attribute Blue BR Attribute ▸
    Data ▸
    ◂ WorkflowModelElement Blue BR WorkflowModelElement ▸
    ◂ EventType complete EventType ▸
    ◂ Timestamp 2011-01-07T14:15:00.000+05:30 Timestamp ▸
    ◂ Originator Dyer Originator ▸
AuditTrailEntry ▸
```

Figure 3. Emerlad Dyeing unit, MS-Access database corresponding MXML log

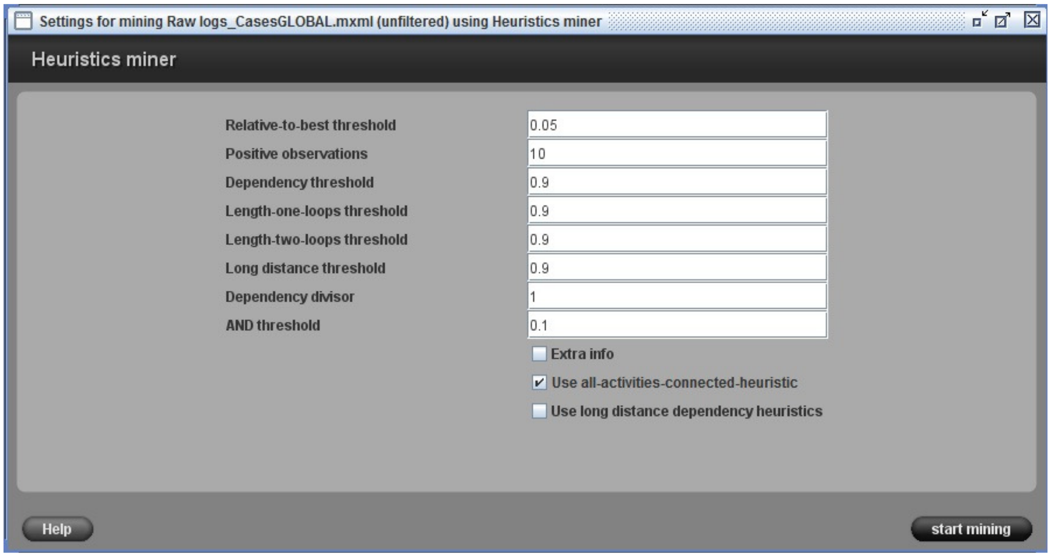
4. HEURISTICS MINER ALGORITHM

The HM algorithm focuses on the control flow perspective and generates a process model in the form of a Heuristics Net for the given event log. The formal approaches like the alpha algorithm i.e. an algorithm for mining event logs and producing a process model, presupposes that the mined log must be complete and there should not be any noise in the log. However, this is not practically possible. Also, this algorithm does not make use of any frequency information i.e. frequency of various dependencies of the tasks in an event log, which can be quite useful in situations of noise. Readers can refer [19] for detailed reading about the alpha algorithm including its limitations. Therefore, the HM algorithm was designed to make use of a frequency based metric and so it is less sensitive to noise and the incompleteness of logs. In the next section the basic concept of this algorithm are explained.

4.1 The Heuristics Mining plug-in

In this section, introduced the Heuristics Mining plug-in implemented in the ProM framework. Hence, first it explained the parameters and then the application of dyeing process pertaining to the Emerald dyeing unit will be discussed.

The discovery of a process model underlying an event log with the help of the HM is based on some parameters. Different values given to these parameters produce a different output, which can be analyzed to obtain meaningful conclusions. Figure 4 shows these parameters and their default values. The different parameters available in the HM are: All-events-connected-heuristic, Dependency Threshold, Dependency divisor, AND Threshold, Positive observations, Relative-to-best Threshold, Length-one-loops Threshold, Length-two-loops Threshold, Long distance threshold, Long distance dependency heuristics and Extra information.


Settings for mining Raw logs_CasesGLOBAL.mxml (unfiltered) using Heuristics miner

Heuristics miner

Relative-to-best threshold	0.05
Positive observations	10
Dependency threshold	0.9
Length-one-loops threshold	0.9
Length-two-loops threshold	0.9
Long distance threshold	0.9
Dependency divisor	1
AND threshold	0.1

☐ Extra info
☑ Use all-activities-connected-heuristic
☐ Use long distance dependency heuristics

Figure 4. Parameter of the Heuristics Miner Algorithm

4.2 Experiments with the Heuristics Miner Algorithm

In this section, experiments with the HM algorithm using dyeing process in dyeing unit processes from Emerald dyeing units were illustrated. The algorithm was used to generate process models for these processes. Through these experiments mainly focus on the evaluation of process models using HM and DWS mining algorithms.

Till now the HM algorithm was tested using benchmark artificial data in [19] and it was found that the algorithm is till date one of the most robust algorithms for event logs containing noise. Therefore, it would also like to determine if the performance of the algorithm on event logs is similar to its performance on the benchmark material. These experiments on this material were done with the default parameter values so the effect of different parameter settings on the output of the algorithm will also be studied in our experiments. For all our experiments database tables converted to MXML logs will be used.

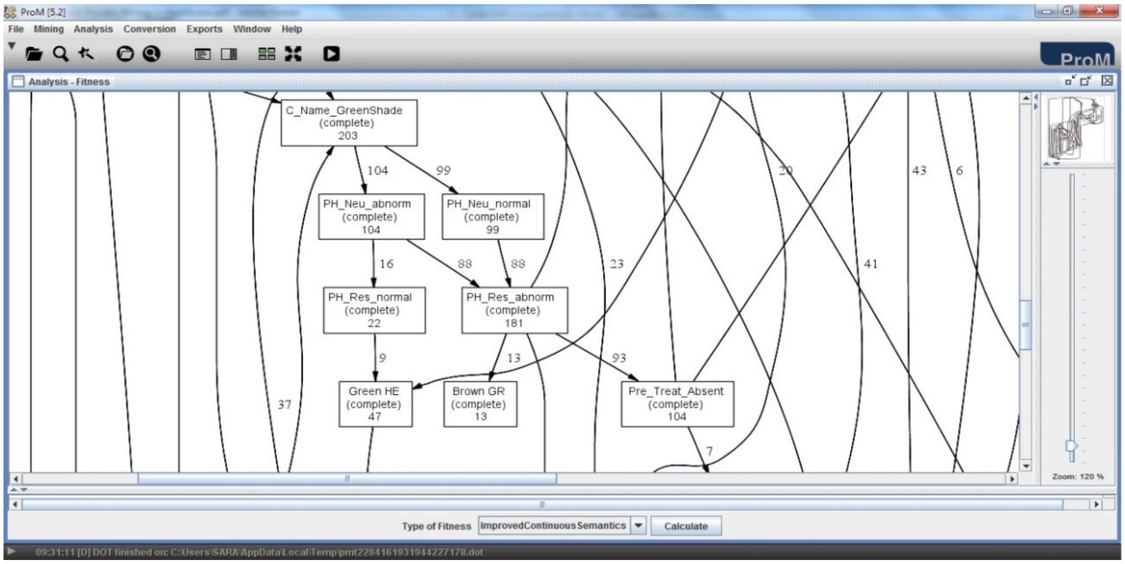
As already mentioned the purpose of the following experiments is to get insights into the dyeing processes by analyzing their process models generated by the HM algorithm. We describe our experiments under the heading *Illustration* and each illustration contains experiments conducted with a specific purpose. Illustration 1 describes the output of the HM algorithm and discusses the problems with it. Illustration 2 describes experiments performed on logs obtained after applying some filtering mechanisms. These experiments also show the effect of different parameter settings on the output of the HM.

4.2.1 Illustration 1

This log has 203 process instances (PIs) and 61 different ATEs. These PIs represent different 'combination paths' followed by different shades i.e. these PIs has different shades of colours i.e. one shade has different colour mix combination leads to another. Each event in a PI is also a combination of pretreatment, prePHtest and colour mixers such as Yellow FG, Turquoise G, Blue BR, etc.

The algorithm was applied with default parameter settings and the output of this experiment is shown in Figure 5. As shown in the figure, the screen is divided into two parts by a separator bar. On the right hand side the structure of the complete process model can be seen and the left hand side shows a part of this complete process model. Following observations were made from this process model:

- The process model has a complex spaghetti-like structure.
- Presence of different activities like *PH_Neu_abnorm*, *PH_Neu_normal, Pre_Treat_absent, Green HE, Brown GR,* etc.
- The Improved Continuous semantics fitness of the model is 0.657


ProM [5.2]
File
Mining
Analysis
Conversion
Exports
Window
Help
ProM
Analysis - Fitness
C_Name_GreenShade (complete) 203
104
99
PH_Neu_abnorm (complete) 104
PH_Neu_normal (complete) 99
16
88
88
23
20
43
6
41
PH_Res_normal (complete) 22
PH_Res_abnorm (complete) 181
9
13
93
37
Green HE (complete) 47
Brown GR (complete) 13
Pre_Treat_Absent (complete) 104
7
Zoom: 120 %
Calculate

Type of Fitness: ImprovedContinuousSemantics

```
09:31:11 [D] DOT finished on: C:\Users\SARA\AppData\Local\Temp\pmt2284161931944227178.dot
```

Figure 5. Dyeing Process Model for Emerald Dyeing Units OneShade logs, fitness = 0.657

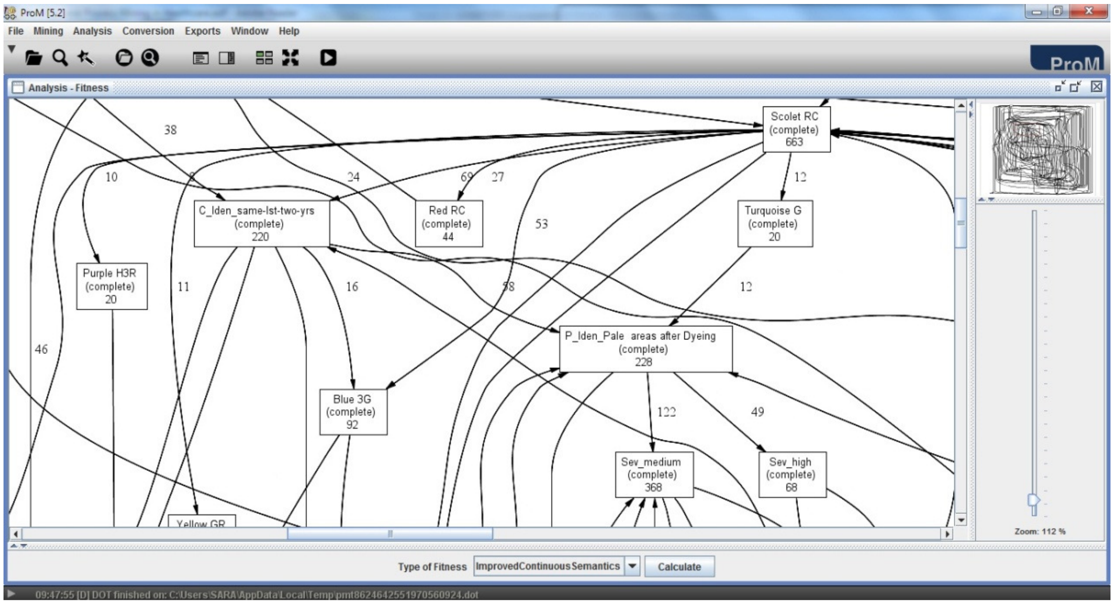
Before analyzing these observations, we would also like to mention the experiment with another dyeing log. The log for this experiment consists of information about color mix or treatments given to various shades. It has 683 PIs and 70 different ATEs.

These PIs represent different 'color mix or treatment paths' followed by different shades i.e. these PIs show for different shades the order in which their one color mix or treatment is followed by another. Each event in the PI is a treatment, for example, *C_Name_GreenShade, PH_Neu_normal, Brown GR and Pre_Treat_Absent*, etc. The output of the algorithm applied with default parameter settings is shown in Figure 6.

- Presence of dangling activities like *Purple H3R, Turquoise G*, etc.
- Besides being complex, the process model misses many dependencies. For instance, it can be seen from the dependency graph that the activity *Purple H3R* is registered in the log 20 times but the process model captures only 10 its incoming connections. The remaining 10 connections are missing. Similarly, the process model does not capture the connection for the activity *Turquoise G has 20 and produce 12, so 8 connections are missing*, etc.
- The Improved Continuous semantics fitness of the model is 0.619

Other observations that were made for this process model are similar to the one made for the first experiment with the combinations log. From the Figure 6, it is apparent that the process model is very complex. Dangling activities and missing dependencies were also observed. Similar results were obtained for many other event logs from Emerald Dyeing units OneShade logs.

Based on the characteristics of the dyeing unit, observations from the Emerald Dyeing units OneShade logs and the understanding of the HM algorithm, it is attempted to analyze the observations made for these two experiment logs.



Type of Fitness: ImprovedContinuousSemantics

```
09:47:55 [D] DOT finished on: C:\Users\SARA\AppData\Local\Temp\pmt8624642551970560924.dot
```

Figure 6. Dyeing Process Model for Emerald Dyeing Units ThreeShade logs, fitness = 0.619

- The complexity of the process models can be attributed to the *uniqueness of cases* i.e. shades in the dyeing domain. Each shade represents a unique and distinct case depending on the specific conditions like previous dyeing process history, responses to certain treatments and various other factors. It is difficult to say that for example, 10 shades have a combination say *A* always has same combinations thereafter. Depending on the specific condition of a shade, the shade may affected by from different combinations following the combination *A* and hence receive different color mix or treatments. This illustrates heterogeneity of shades in the dyeing domain. Owing to this heterogeneity and uniqueness of shades present in the event log, the HM algorithm results into a complex process model.

In spite of the addition of the unique start and end events, some dangling activities are found in the process model. The reason can be the *presence of noise.* It is quite possible that due to errors some unwanted events are inserted or some relevant events are missed from the event log. This may result into missing further connections between any dangling activities with other activities in the log. This shows some possible problem or issue from the algorithm point of view. Though registered 20 times in the log, the process model captures only 10 of its outgoing connections. The remaining 10 connections for this activity are not found in the model. This can be again be the result of noise in the log. It is possible that erroneous insertion or deletion occurred in the event log resulting into loss of relevant connections for some activities. Besides missing connections, the process model also misses some events which are registered in the log but are not captured in the model. This may be because they are low frequent events and these are left out in the model because they do not fulfill certain parameters of the algorithm.

- The low frequent events found in the log may be due to noise or dyeing exceptional cases. But it can also not be ignored that the low frequent events are common in the very critical shades of any dyeing organization. The course of color mix or treatment constantly needs to be determined for such shades. In this situation it is difficult to find any standardized process. Therefore, the HM algorithm does not generate simpler models for such flexible processes. But it is also equally true that the HM algorithm cannot distinguish low frequent behaviour from noise.
- Fitness is a quality measure indicating the gap between the behaviour actually observed in the log and the behaviour described by the process model. It gives the extent to which the log traces can be associated with execution paths specified by the process model [20]. The second process model in the illustration above model has a poor fitness value indicating that most of the log traces are not successfully parsed by the mined process model. This may be because of the presence of noise resulting into dangling activities and missing connections. It is also possible that the parameter settings do not discover all connections.

From this analysis we can say that the complexity of the process model, presence of dangling activities and other problems mainly stem from the underlying investigated dyeing process. Therefore, it can make an attempt to obtain understandable process models by varying the parameter settings of the HM algorithm. In the next illustration, the impact of varying the parameter settings on the output of the HM is discussed.

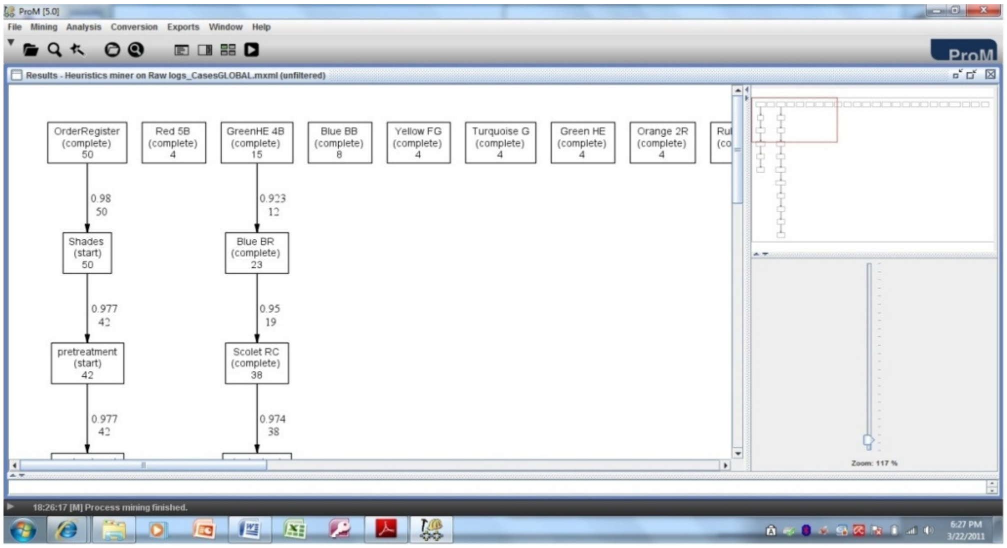
4.2.2 Illustration 2

For the experiments described here, same logs as used in Illustration 1 are used. The process models obtained from different parameter settings are not shown due to the limitation of space, but the key observations are mentioned below:

- To generate the main behaviour of the process, set high values for parameters like Positive Observations, Dependency threshold, Length-one-loops threshold and Length-two-loops threshold. As already indicated in Section 4.1, higher values of these parameters generate main behaviour of the process. The resulting process models obtained with these parameter settings were less complex as compared to model in Figure 6 but a lot of dangling activities and missing connections are still observed.

- When the event log for combinations was mined with default parameter settings except using the *all-activities-connected heuristic*, totally disconnected activities were observed. In presence of the unique start (*ArtificialStartTask)* and unique end (*ArtificialEndTask)*, the activities are connected only to the start and end tasks, and a lot of dangling activities are also present. Besides, these connected activities, a lot of disconnected activities are also found. It is shown in the figures 7 and 8 for these two parameters
- When the event log for color mix or treatments was mined with default parameter settings except using the *allactivities-connected heuristic*, although a better model was obtained compared to models in Figure 5 and 6 in terms of simplicity and ease of understanding but a lot of dangling activities, totally unconnected activities and missing connections are also found. These models contain a lot of low frequent events. As already mentioned that these may be due to noise or actually low frequent event. So, if the log is cleaned from such low frequent events which are also shown as dangling activities and then the HM algorithm is applied, simpler and complete models may be obtained. From the analysis in Illustration 1 and 2, we can say that the complexity of the process model, presence of dangling activities and other problems mainly stem from the underlying investigated dyeing process. The input to the algorithm is a dynamic, flexible and less structure event log. The HM does not simplify the output from the complex input it receives in the form of logs. If the parameter *all-events-connected heuristic* is not used then, it get simpler models but the behaviour represented in this model is very much incomplete. The focus and emphasis in a domain like the dyeing is on the simplicity and the completeness of the model. Therefore, some techniques must be found out to retrieve simpler and easier to understand process models.
- One of the ways can be abstracting the input event log in order to retain only some desired portions of the log. Below it is given that a brief overview of how abstraction can be achieved. Abstraction is a relative process. It highly depends on what results one would like to obtain. For example, from the combinations log, the interrelationship between various combinations may be of interest or it would be interesting to find combinations found in shades of certain group. An event log that records information about shades combinations and color mix or treatments would be interesting to discover what color mix or treatment procedures are followed for combinations of a specific category. Based on the desired results an event log can be abstracted. In the ProM framework, abstraction can be done in the following ways:
- Using different filters: The ProM framework offers a variety of filters. For example, the *Event log filter* enables the selection of only desired activities, the *Enhanced event log filter* enables a user to specify relative percentage of an activity in the entire log whereas activities performed by a particular originator can be selected using the *Originator log filter* etc. Based on a user's requirements these filters can be used to abstract an event log. Further, the instances satisfying the filtering criteria can be then exported to a new log file using the Export plug-ins.

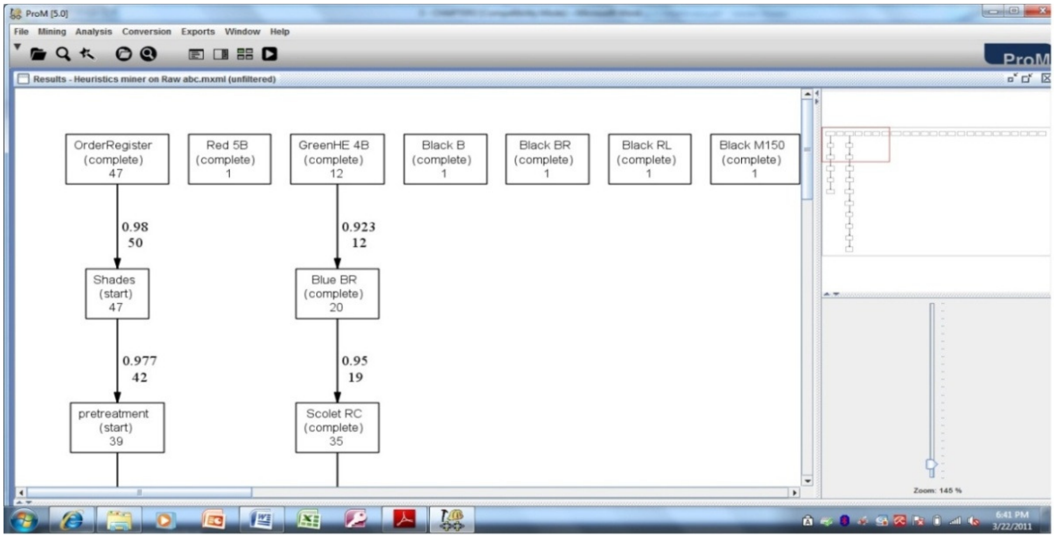
The process model can focus on the main behavior of the process and the log can be filtered from certain low frequent behaviour. It should however be noted that by abstraction a smaller part of the entire log is used and the behaviour that do not satisfy the criterion is lost. But it is equally true that a logically abstracted log makes it easy to focus on investigating the obtained process model underlying a dyeing process. It also becomes easy to investigate how different parameters of the algorithm interact to produce a certain output. The experiments illustrated now onwards are done on abstracted logs as we intend to provide the HM with simple input logs in order to achieve simple and nicer process models unlike the complex models retrieved in the above experiments.



Results - Heuristics miner on Raw logs_CasesGLOBAL.mxml (unfiltered)

18:26:17 [M] Process mining finished.

Figure 7. Dependency graph for combinations log, *all-activities-connected heuristic=false*



Results - Heuristics miner on Raw abc.mxml (unfiltered)

Figure 8. Dependency graph for combinations log with unique start and end points, *all-activitiesconnected heuristic=false*

4.2.3 Summary: Experiments and Observations

The experiments conducted in the previous subsections were aimed at obtaining insights into the dyeing processes and evaluating the HM algorithm for these processes. The effect of parameter settings on the output of the HM algorithm summarized with observations and findings from these experiments:

1. The models for the dyeing processes contain complex spaghetti-like structures. The algorithm generates models for the input it is provided. In this case the inputs are the various dyeing logs. These logs illustrate the flexible and less structured processes of the dyeing domain. Owing to these characteristics of the domain, the focus of this paper assignment was to obtain simpler models that could be analyzed for extracting meaningful information about the underlying processes concerning various shades. The HM however does not simplify its output for the complex input it is provided. In unstructured processes the control flow paths do not have fixed form and this flexibility leads to the complex process models.

2. Problems like dangling activities, missing connections, missing activities are found in almost all the experiments done with the dyeing logs. Many of these problems can be attributed to the presence of noise or low frequent behaviour. Low frequent behaviour in the dyeing domain represents exceptional dyeing process cases. These exceptional dyeing process cases can be of great interest but while capturing them the algorithm can also capture noise. It is unable to distinguish between noise and low frequent behaviour which puts a question mark on the behaviour captured in the mined model.

3. It was also seen that the algorithm generates different models for different parameter settings. Sometimes the settings produce the desired clean model but sometimes the existing activities and connections in the model are also lost. It is also observed that the number of parameters affecting the output of the algorithm is too large. This dependency of the algorithm on its various parameters leads to further problems like missing dependencies or activities, dangling activities and confusion whether which process model to trust as well as the large number of parameters makes it difficult and confusing for a user to obtain his desired model. Moreover, it is also not possible to reach to an optimum parameter setting for all the event logs.

4. It was also discovered that the activities in the dyeing log cannot be always characterized as clear AND/ XOR join/ splits. They sometimes belong to both of them. This typical characteristic of dyeing domain is not captured in the heuristics net provided by the algorithm.

5. It was also observed that when the parameter *all-activities-connected heuristic* is not used, the algorithm can generate better and simple models for some logs but this cannot be generalized. For example, in case of combinations log without a unique start and end point, the process model obtained was just a collection of unconnected activities. Whereas, for color mix or treatments log without start and end point, though the model consisted of dangling and unconnected activities it was simple to understand and conveyed some information about the underlying process. The structure of this model was not huge and confusing. But as already mentioned this parameter does not give desired and informative models for all logs. So, it cannot be concluded whether not using the *all-activities connected heuristic* is a good choice. Based on the above observations and findings, we conclude that the HM may not be the appropriate algorithm to gain insights into the processes of the dyeing domain.

6. It also convinced that the heuristics net representation used in the HM is not suitable for dyeing domain because it is unable to represent mixed AND and XOR situations discussed before. Therefore some alternate process model representations overcoming these limitations must be explored. In this context, we found the Disjunctive Workflow Schema (DWS) algorithm interesting because it not only gives visual process models

but also rules which represent implicit behavioural pattern present in the log. This algorithm is implemented in ProM as DWS mining plug-in. It is elaborated and investigated this algorithm in the next section.

5. THE DISJUNCTIVE WORKFLOW SCHEMA ALGORITHM

The DWS approach attempts to provide insights into a process whose enactment is constrained by some kind of rules, possibly involving information that is beyond the pure execution of activities [21]. It accepts an event log as an input and finds behavioural patterns in the log in form of rules known as discriminant rules. These rules are representations of constraints in the event log, which otherwise go unnoticed and undiscovered by other mining algorithms. This plug-in calculates these rules over projected traces in the log and further they are used for partitioning the event log into variants. These rules are defined as follows:

"A discriminant rule is a rule of the form, [a1….ah]-/->a such that

- *The frequency of [a1…..ah] and [aha] in the log is over a given threshold sigma, i.e. they are both highly frequent and,*
- *The frequency [a1….aha] is below a given threshold gamma, i.e. it is lowly frequent."*

For example, consider a discriminant rule: *a*, *b* -/-> *d* where *a*, *b* and *d* are activities in an event log. The rule states that the frequency of the activity sequence *ab*, and *bd* of the activities *a*, *b* and *b*, *d* respectively is above a threshold value (called *sigma*) specified by the user, but the frequency of the activity sequence *abd* of the tasks *a*, *b* and *d* together is less than a user specified threshold (called *gamma*). This approach is discussed in simple words as below:

1. Input: Event log of a process
2. First step is to discover the overall workflow schema
3. Iteratively refine this schema by:
   a. Finding discriminant rules
   b. Cluster the traces characterized by these rules.
4. Use some mining algorithm to generate process models for these clusters.
5. The overall workflow schema then is the set of all the process model variants generated
   in step 4.

The DWS plug-in was designed to discover both the control flow of a given process and the interesting global constraints which presents a refined view of the process. Traditionally the control flow perspective prescribes only the local constraints and misses out on the global ones.

The local constraints are in form of relationships of precedence of tasks in a process, viz., an AND-join activity is executed only after all its predecessors are completed etc. Global constraints are richer in nature and their representation strongly depends on the particular application domain of the modeled process. The basic idea of the DWS approach is to first derive from the event log an initial workflow schema whose global constraints are left unexpressed and then to stepwise refine it into a number of specific schemas, each one modeling a class of trace having the same characteristics with respect to global constraints [21]. In the next section an example illustrates the working of the DWS algorithm.

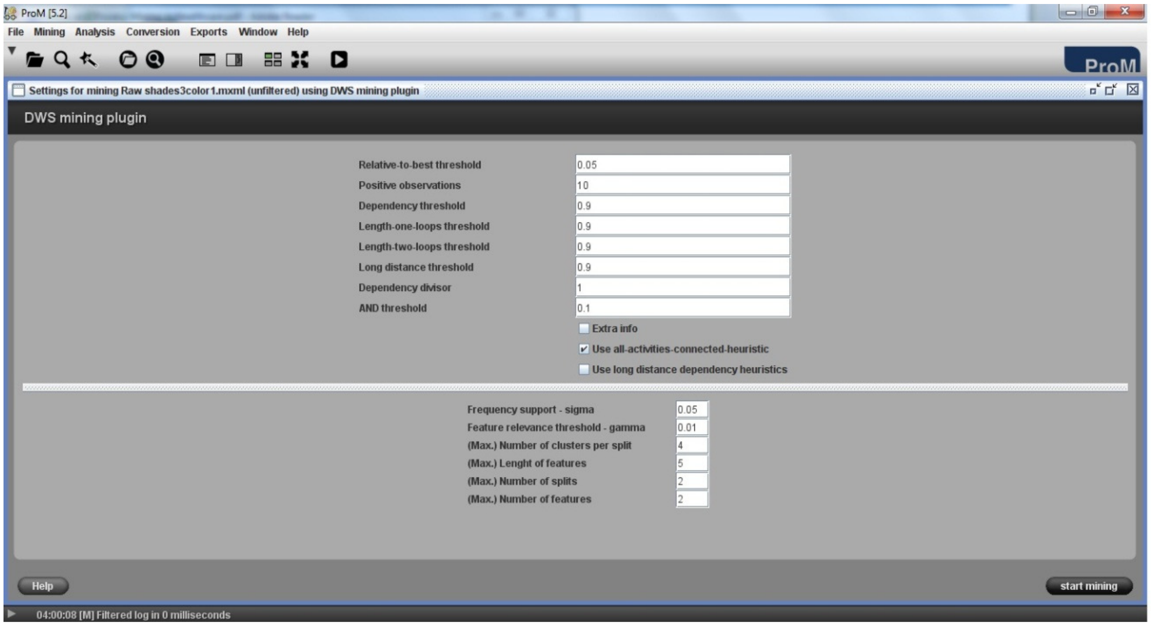
5.1 The DWS plug-in

Figure 9 shows a screenshot of the DWS plug-in. Two frames divided by a separator can be seen in this figure. The parameters for the DWS plug-in are located in the bottom frame and in the top frame as we can recall are the parameters from the Heuristics Mining plug-in. The HM is used to construct the initial workflow schema and the process models from the traces characterized by different discriminant rules generated by the DWS algorithm. In spite of the limitations of the HM, the DWS plug-in based on the HM was chosen because the HM is robust to noise and imbalance. Therefore, to study an algorithm that provides some alternate process model representations. The DWS besides providing the process model in form of the dependency graph also provides discriminant rules which convey behavioural information about the process.

In Figure 9 it can be seen that the values of the frequency thresholds: *gamma* and *sigma* can be specified by a user. Now as it is known that the DWS algorithm first derives an overall workflow schema of the underlying process, and then this schema is iteratively refined and clustered using the *k*-means clustering algorithm [22], on the basis of these rules the number of required refinements can also be specified by the parameter *Number of splits*. The parameter *Number of clusters per split* specifies the maximum number of *k* clusters to be used in the K-means algorithm. The number of rules to be mined as well as their length can be specified through the parameters: *Number of features* and *Length of features* respectively. The default values of all these parameters can be seen in the Figure 9. Figure 10 shows the results of mining the combinations log using the DWS approach. The global workflow schema is represented by R. Two discriminant rules are discovered for this initial workflow schema and the process model characterizing these rules is also shown on the frame at the right hand side. Let us understand one of these rules. Consider the following rule:

Blue BR, Red RC, Yellow 5GL-/-> Scolet RC

The rule can be interpreted as: ”the tasks in the dyeing process such as *Blue BR, Red RC* and *Yellow 5GLr* occur in this sequence more than 5% i.e. the value of sigma is 0.05 in the event log as well as the tasks *Yellow 5GL* and *Scolet RC* also occur more than 5% in the log, but their combination in the same order occurs less than 1% in the event log. Although the mined process model allows for this behaviour, in context of the dyeing it can be said that in the event log the shades affected by the combinations: *Blue BR, Red RC* and *Yellow 5GL* and the shades affected by the combinations *Yellow 5GL, Scolet RC* are highly frequent i.e. found in 5% of the log traces but the number of shades affected by all three combinations in the same order are low frequent found only in 1% of the log traces.



Settings for mining Raw shades3color1.mxml (unfiltered) using DWS mining plugin

DWS mining plugin

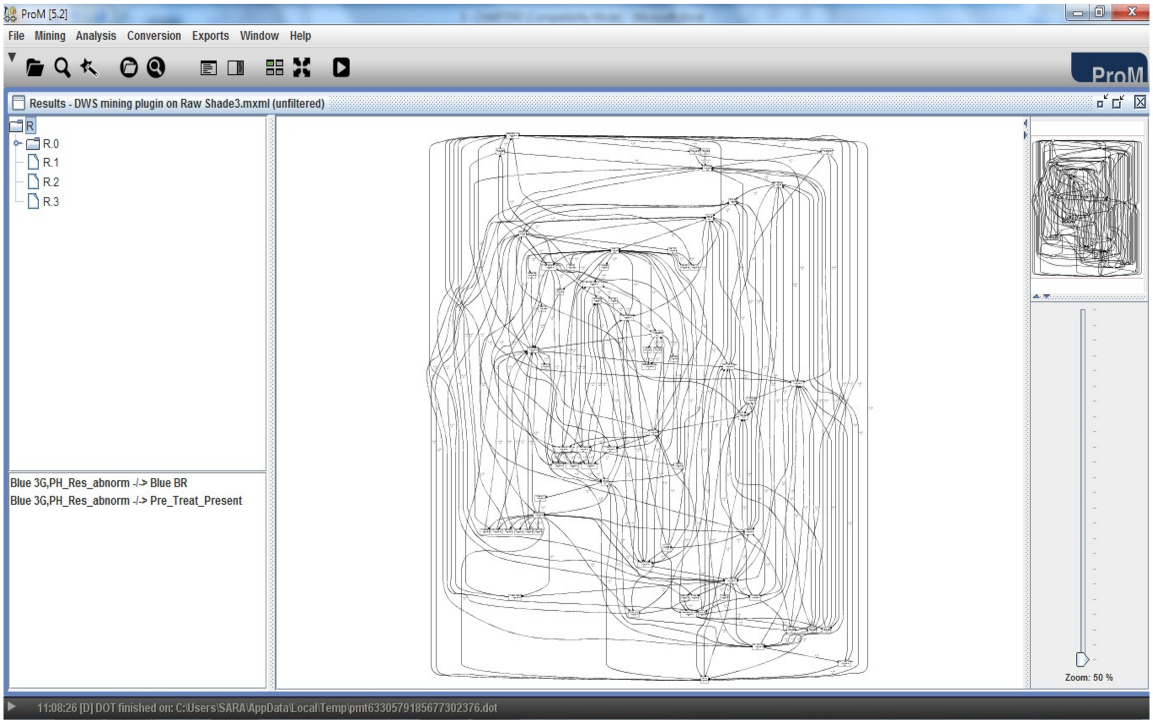
Relative-to-best threshold	0.05
Positive observations	10
Dependency threshold	0.9
Length-one-loops threshold	0.9
Length-two-loops threshold	0.9
Long distance threshold	0.9
Dependency divisor	1
AND threshold	0.1

□ Extra info
☑ Use all-activities-connected-heuristic
□ Use long distance dependency heuristics

Frequency support - sigma	0.05
Feature relevance threshold - gamma	0.01
(Max.) Number of clusters per split	4
(Max.) Lenght of features	5
(Max.) Number of splits	2
(Max.) Number of features	2

```
04:00:08 [M] Filtered log in 0 milliseconds
```

Figure 9. Parameters for the DWS plug-in



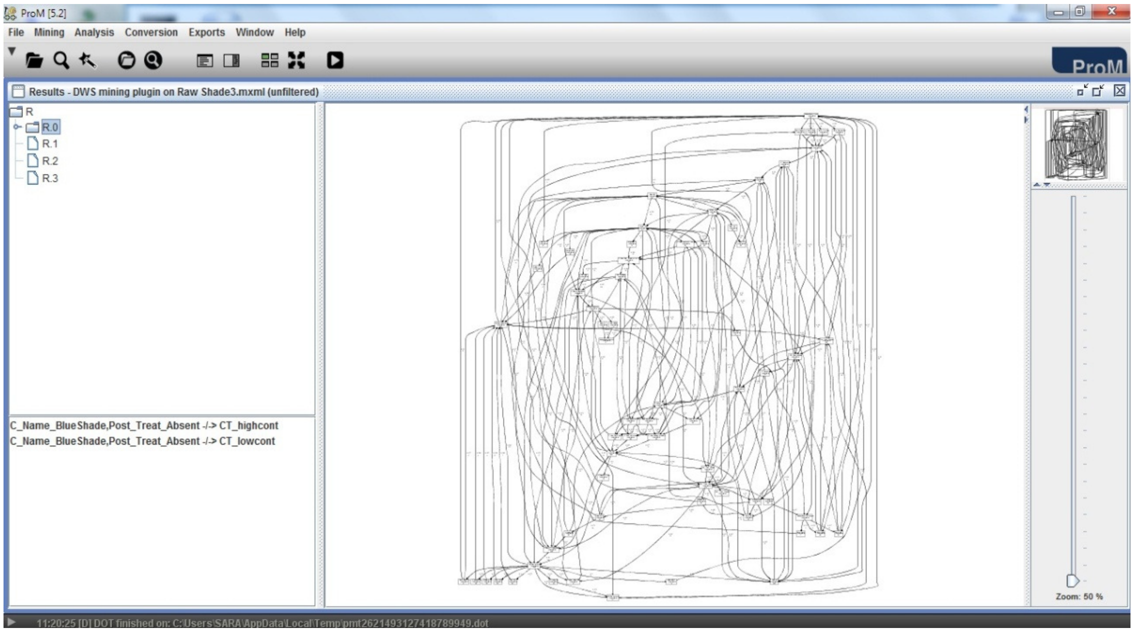
Results - DWS mining plugin on Raw Shade3.mxml (unfiltered)

Blue 3G,PH_Res_abnorm -/-> Blue BR
Blue 3G,PH_Res_abnorm -/-> Pre_Treat_Present

```
11:08:26 [D] DOT finished on: C:\Users\SARA\AppData\Local\Temp\pmt6330579185677302376.dot
```

Figure 10: Discriminant rules and a process model as output of the DWS plug-in

This initial workflow schema R is further refined in two clusters: R.0 and R.1. Figure 11 and 12 shows the variants of the process model obtained from refining R and characterized by the discriminant rules.

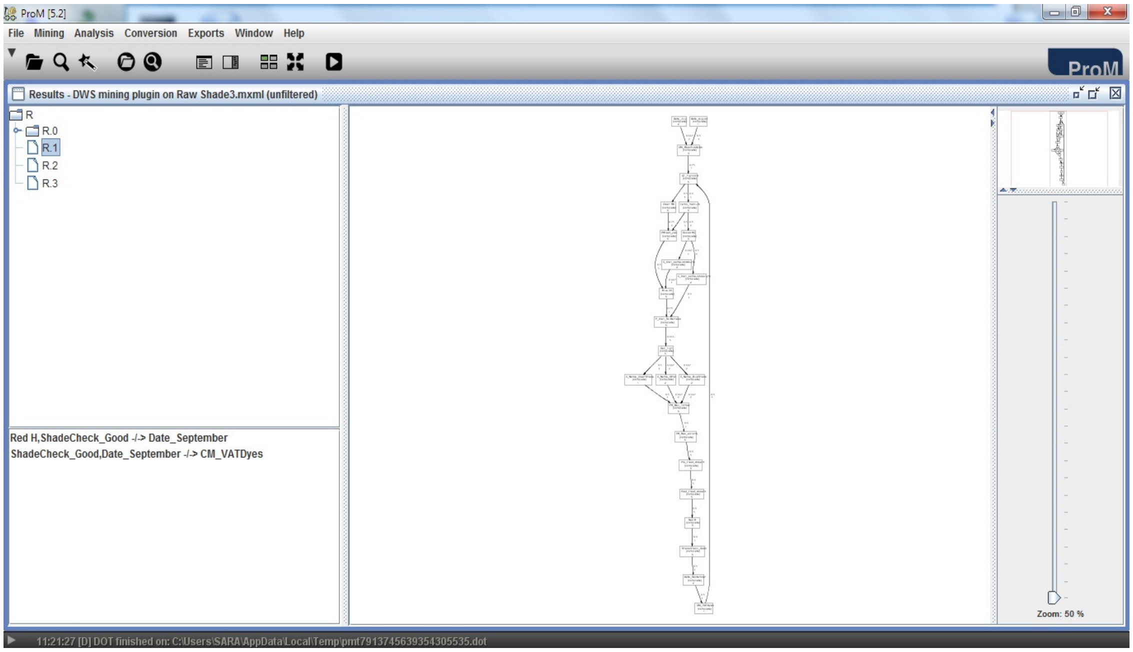


Results - DWS mining plugin on Raw Shade3.mxml (unfiltered)

C_Name_BlueShade,Post_Treat_Absent -/-> CT_highcont
C_Name_BlueShade,Post_Treat_Absent -/-> CT_lowcont

11:20:25 [D] DOT finished on: C:\Users\SARA\AppData\Local\Temp\pmt2621493127418789949.dot

Figure 11: The cluster R.0



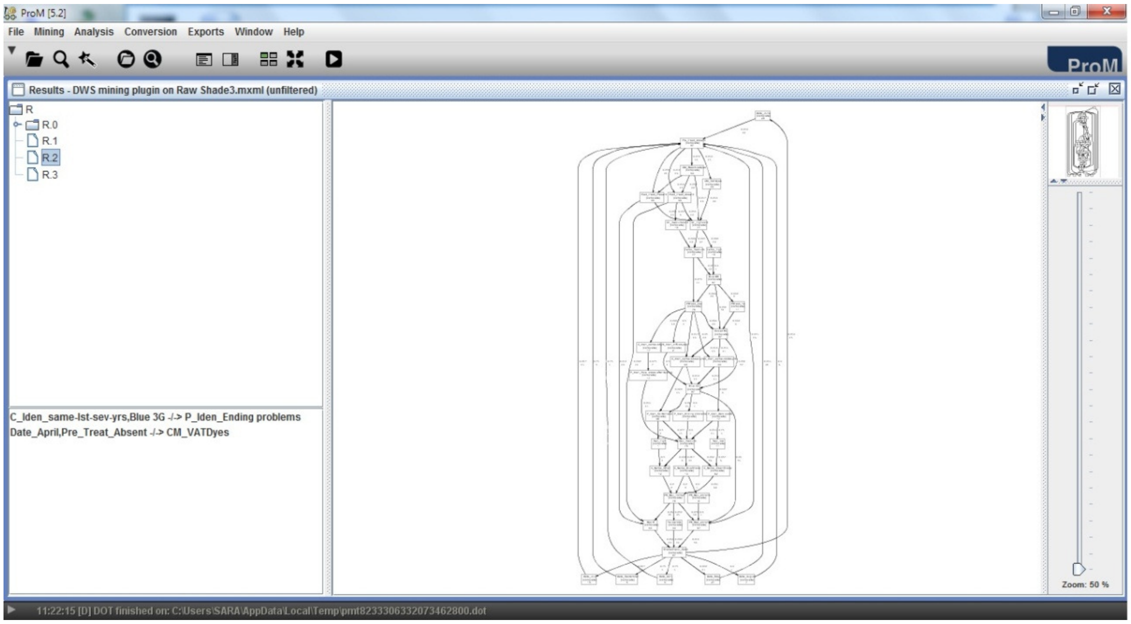
Results - DWS mining plugin on Raw Shade3.mxml (unfiltered)

Red H,ShadeCheck_Good -/-> Date_September
ShadeCheck_Good,Date_September -/-> CM_VATDyes

11:21:27 [D] DOT finished on: C:\Users\SARA\AppData\Local\Temp\pmt7913745639354305535.dot

Figure 12: The cluster R.1

Also the workflow schema R is further refined in two clusters: R.2 and R.3. Figure 13 and 14 shows the variants of the process model obtained from refining R and characterized by the discriminant rules.

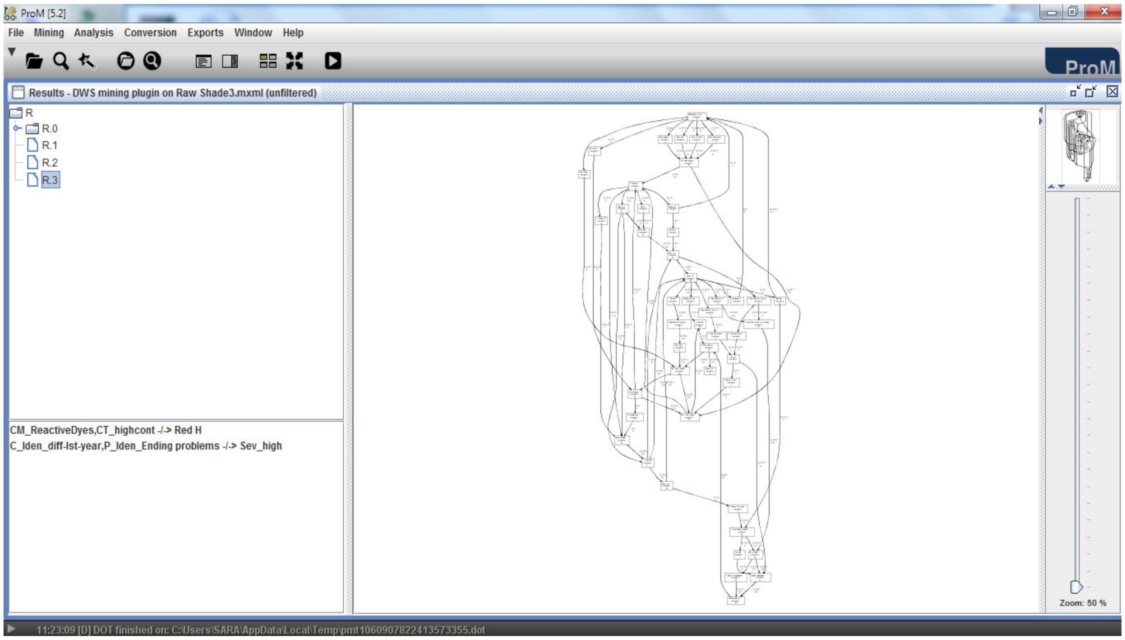


Results - DWS mining plugin on Raw Shade3.mxml (unfiltered)

C_Iden_same-Ist-sev-yrs,Blue 3G -/-> P_Iden_Ending problems
Date_April,Pre_Treat_Absent -/-> CM_VATDyes

```
11:22:15 [D] DOT finished on: C:\Users\SARA\AppData\Local\Temp\pmt8233306332073462800.dot
```

Figure 13: The cluster R.2



Results - DWS mining plugin on Raw Shade3.mxml (unfiltered)

CM_ReactiveDyes,CT_highcont -/-> Red H
C_Iden_diff-Ist-year,P_Iden_Ending problems -/-> Sev_high

```
11:23:09 [D] DOT finished on: C:\Users\SARA\AppData\Local\Temp\pmt1060907822413573355.dot
```

Figure 14: The cluster R.3

These discriminant rules express the behavioral patterns amongst the log activities. If algorithms like the α-algorithm or the HM were used on a log, only process models would have been generated. These process models depict the local constraints in form of control flow. The DWS approach also expresses the global constraints in the form of discriminant rules. It is found that the strength of the DWS approach that it discovers both, the global as well as the local constraints from an event log. As opposed to the complex and huge process models generated by the Heuristics Mining plug-in, the DWS plug-in generates simpler and easy to comprehend process models for the sub-clusters like R.0, R.1 etc. But as the DWS uses the HM to generate the process models, it inherits the problems of the HM algorithm too. The user might also be prompted to experiment with different parameter settings of the HM, which can be quite tricky as the changes in the parameters may not always lead to desired results. However, the advantage of the DWS plug-in over the HM algorithm is that smaller process models are obtained based on some behavioural pattern specific to particular process model variants of the entire schema. Hence, even if complex models are obtained for sub clusters i.e. in this case used a small and filtered log therefore, obtained simple models at the sub clusters. Therefore, the rules generated by the plug-in provide some kind of behavioural information about the process. These behavioural patterns serve two purposes: first they are used as the basis of clustering the traces and second they represent some kind of information about the activities in the event log. Besides these advantages the DWS has certain limitations discussed in the next section.

5.2 Observations

This section mentions some of the limitations that were discovered in the DWS plug-in:

1. The discriminant rules are not simple to understand at a first look. For example, if we derive a rule like “Blue BR, Red RC, Yellow 5GL -/-> Scolet RC” from the dyeing processing data and present it to the stakeholders they would find this rule difficult to comprehend as it deals with parameters associated with frequency. The stakeholders at large are not technical people, they can be dyers, and other staff from the dyeing who would like to benefit from this rule. But to understand and use the knowledge represented by this rule, they have to understand the threshold values- sigma and gamma, otherwise it is difficult for them to understand the rule.

2. The discriminant rules deal only with neighbouring tasks. This is a shortcoming as it gives the relationship of a task only in context of its neighbour and then its relationship with non neighbouring task is neglected. It seems like loss of information or lack of information as a task may also be related to other tasks that are not its neighbours. In context of dyeing, this incomplete information may be dangerous in place of being beneficial. If the relationship of some combination task is known only in context of combinations that directly precede or follow it, and no information about its relationship with other combinations is given, this information is not useful for the stakeholders from the dyeing domain.

3. Although the workflow schemas which are iteratively generated and guided by the notions of completeness and soundness, it is difficult to know the relative importance of different rules that are generated. Though these rules are ordered based on their importance but their importance is not quantified. The rules are not accompanied by some metric that depicts their importance in comparison with the other generated rules. It is only known that from top to bottom the importance decreases but how much is unknown. So though, the plug-in orders the rules, it lacks the quantification of their importance.

4. Currently the DWS mining plug-in uses the Heuristics Mining plug-in. Therefore, the problems faced with the HM in context of dyeing will also have to be dealt while using the DWS plugin.

6. CONCLUSION

In this paper, it is used the HM algorithm in the dyeing process and illustrated with experiments. Section 4.2.3 listed the limitations of this algorithm and this formed the motivation for investigating the DWS approach in Section 5.1. It was found that the strength of the DWS plug-in lies in its discovery of global and local constraints. The global constraints are discovered in form of discriminant rules and the local constraints are comprised in the variants of the process model represented by the various clusters. Limitations of this approach were stated in Section 5.2. Therefore it seems that we need to look at some other alternate process model representations which overcome the limitations of both the HM as well as the DWS algorithm.

REFERENCES

[1] Zeromskiego. U, "Fibres and Textiles in Eastern Europe", October Vol 15, BO. 4(63), December 2007.

[2] W.M.P. van der Aalst and A.J.M.M. Weijters. Process mining: a research agenda. In Computers in Industry 53, Elsevier B.V., 2003.

[3] R. Agrawal, D. Gunopulos, and F. Leymann. Mining Process Models from Workflow Logs. In Sixth International Conference on Extending Database Technology, pages 469–483, 1998.

[4] J.E. Cook and A.L. Wolf. Discovering Models of Software Processes from Event-Based Data. ACM Transactions on Software Engineering and Methodology, 7(3):215–249, 1998.

[5] J.E. Cook and A.L. Wolf. Event-Based Detection of Concurrency. In Proceedings of the Sixth International Symposium on the Foundations of Software Engineering (FSE-6), pages 35–45, 1998.

[6] J.E. Cook and A.L. Wolf. Software Process Validation: Quantitatively Measuring the Correspondence of a Process to a Model. ACM Transactions on Software Engineering and Methodology, 8(2), pages 147–176, 1999.

[7] J. Herbst. A Machine Learning Approach to Workflow Management. In Proceedings 11th European Conference on Machine Learning, volume 1810 of Lecture Notes in Computer Science, pages 183–194. Springer-Verlag, Berlin, 2000.

[8] J. Herbst. Dealing with Concurrency in Workflow Induction. In U. Baake, R. Zobel, and M. Al-Akaidi, editors, European Concurrent Engineering Conference. SCS Europe, 2000.

[9] J. Herbst and D. Karagiannis. Integrating Machine Learning and Workflow Management to Support Acquisition and Adaptation of Workflow Models. In Proceedings of the Ninth International Workshop on Database and Expert Systems Applications, IEEE, pages 745–752, 1998.

[10] J. Herbst and D. Karagiannis. An Inductive Approach to the Acquisition and Adaptation of Workflow Models. In M. Ibrahim and B. Drabble, editors, Proceedings of the IJCAI'99 Workshop on Intelligent Workflow and Process Management: The New Frontier for AI in Business, pages 52–57, Stockholm, Sweden, August 1999.

[11] L. Maruster, W.M.P. van der Aalst, A.J.M.M. Weijters, A. van den Bosch, and W. Daelemans. Automated Discovery of Workflow Models from Hospital Data. In B. Kr¨ose, M. de Rijke, G. Schreiber, and M. van Someren, editors, Proceedings of the 13th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC 2001), pages 183–190, 2001.

[12] L. Maruster, A.J.M.M. Weijters, W.M.P. van der Aalst, and A. van den Bosch. Process Mining: Discovering Direct Successors in Process Logs. In Proceedings of the 5th International Conference on Discovery Science (Discovery Science 2002), volume 2534 of Lecture Notes in Artificial Intelligence, pages 364–373. Springer-Verlag, Berlin, 2002.

[13] G. Schimm. Process Mining. http://www.processmining.de/.

[14] G. Schimm. Generic Linear Business Process Modeling. In S.W. Liddle, H.C. Mayr, and B. Thalheim, editors, Proceedings of the ER 2000 Workshop on Conceptual Approaches for E-Business and The World Wide Web and Conceptual Modeling, volume 1921 of Lecture Notes in Computer Science, pages 31–39. Springer-Verlag, Berlin, 2000.

[15] H. Mannila and D. Rusakov. Decomposing Event Sequences into Independent Components. In V. Kumar and R. Grossman, editors, Proceedings of the First SIAM Conference on Data Mining, SIAM, pages 1–17, 2001.

[16] H. Mannila, H. Toivonen, and A.I. Verkamo. Discovery of Frequent Episodes in Event Sequences. Data Mining and Knowledge Discovery, 1(3), pages 259–289, 1997.

[17] Perspectives on Information Technology for the Dyeing Care Industry. Tunitas group, Workflow Automation. http://www.tunitas.com/ pages/Workflow/ workflow.htm.

[18] Prom Framework: www.processmining.org

[19] M. Dumas, W. M. P. van der Aalst, and A. H. ter Hofstede. *Process-Aware Information Systems: BridgingPeople and Software Through Process Technology*. John Wiley & Sons, Inc. 2005.

[20] A. Rozinat and W. M. P. van der Aalst. *Conformance Testing: Measuring the Fit and Appropriateness of Event Logs and Process Models*. Business Process Management Workshops, pages 163-176, 2005.

[21] G. Greco, A. Guzzo, and L. Pontieri. *Discovering Expressive Process Models by Clustering Log Traces*. IEEE Transactions on Knowledge and Data Engineering 18(8), pages 1010-1027, Aug 2006.

[22] B. Mirkin. *Clustering for data mining: a data recovery approach*. Publisher London: Chapman and Hall/CRC, 2005.

Authors

Saravanan. M.S received B.Sc degree in computer science from Madras University in 1996, the MCA degree from Bharathidasan University in 2001, the M.Phil degree from Madurai Kamaraj University in 2004, M.Tech degree from IASE University in 2005. And now pursuing PhD degree in Bharathiar University. His current research interests include Process Mining, Business Process modeling, Workflow management systems and Exception handling etc. He is an Assistant professor in the Department of Information Technology in VEL TECH Dr, RR & Dr. SR Technical University, Avadi, Chennai, India. M.S. Saravanan has published eleven international publications and presented ten research papers in international and national conferences, having 11 years of teaching experience in various institutions in India.

Rama Sree. R.J received M.S degree in computer science from BITS Pilani University in 1996 and PhD degree in S.P. Mahila University, Tirupati. She is a Reader in Department of Computer Science in Rashtriya Sanskrit University, Tirupati. Dr. Rama Sree has published three books and twenty four international publications and ten national publications, having 17 years of teaching experience.