Workflow Mining

Ameya Murar
aam9527@rit.edu

Chandni Pakalapati
cp6023@rit.edu

Harsh Patil
hpp6869@rit.edu

Priyanka Samanta
ps7723@rit.edu

ABSTRACT

Workflow mining is an important process to extract the workflows in a system and can be used with existing business models or the ones specifically designed around it. The process of workflow mining involves deducing workflows from a given set of event logs. These logs are generated by a business process and usually contain the activity id, session id and a timestamp. Our aim in this project is to improve the workflow mining process in-order to create better decision making from the decision points. A point which can lead to multiple paths in a workflow is called as a decision point. Using hospital log data we plan to identify these decision points and build a decision model from it. We hope this decision model will help in better decision making in business processes.

1. INTRODUCTION

Many enterprise information systems employ workflow management concepts. Generic workflow mining and structured business processes are offered by workflow management systems like IBM and Staffware. To model a workflow, we need expertise in workflow knowledge and deep communication with the workers and their management. In short, workflow mining is not a trivial thing. In this project, we plan to design a workflow decision model which will equip businesses while taking decisions. We work with the dataset of a Dutch Academic Hospital which include real life logs of patients going through the treatment and diagnosis process at the facility. The logs used here are very large in terms of the different domain it covers in the process. We first try to analyze this data and pre-process it to find more homogeneous subsets. The subset we find would then be used to mine the workflow from. This workflow model will help us find the decision points required for our research of improving the decision mining algorithm.

The systematic organization of resources into processes which provides services, modifies materials and assimilates information enabling a repeatative pattern of business activity consists of a workflow. It can be represented as a sequence of events of an entity. This entity could be a group or a person. Workflow is considered as a representation of real work at a high-level perspective. Process mining is a concept which is loosely based on workflow mining. When other approaches fail to provide any formal description of a process, workflow mining techniques are used. Workflow mining consists of three major parts. They are:- Workflow Analysis, Workflow Design and Workflow Enactment. In Workflow Analysis, logfiles are filtered, ordered and compressed to gain further knowledge into the operations of the workflow. In Workflow Design, actions and event logs are monitored whose feedback supports the design. In Workflow Enactment, the results obtained from the operations for triggering further processes are used.

Explicit process model drive contemporary workflow management systems. Creating a workflow design occupies a lot of time and there are cases where the actual process and the process as understood by the management differ. So, techniques which discover workflow models have been implemented. The actual execution of the workflow process is contained in the "workflow log". Process models can be extracted from such logs using alpha algorithm. The output from such process models are represented in petri net format. Arbitrary workflow processes are impossible to be discovered.

Some of the challenges of workflow mining include generating good workflow models from very little information. Workflow logs generally consist of a lot of noise. Noise here refers to incomplete, missing or inconsistent event data. Another challenge is exploiting additional information from little information, which is very tough as less information means more latent attributes which need manual intervention to be unearthed. Also, if the logs are not in English, then it becomes necessary to translate them so as to use them in practice and it makes sense to anyone reading it.

In order to address the above challenges and to implement the concepts listed above, we read many papers to understand the concept behind workflow mining and decided on a strategy to process workflow logs to gain insight into process mining. For this purpose, we selected a dataset from a Dutch hospital in order to understand the flow of organisation within a bigger organisation. The dataset was in dutch language which we translated into English.

In the below section, a thorough review of all the papers read is given. The papers were read so as to gain a complete understanding of the research work done already in this area. After reading the papers, we have devised a 4 step plan of

implementation.

2. RELATED WORK

To get a good understanding of the problem, twelve papers were read. This section summarizes all the papers.

2.1 Event log analysis and pre-processing

We look into a technique to preprocess the event logs which may help us in improving our proposed solution. Event logs as any other data is very important in the process of workflow mining, hence correcting or classifying these logs is an important task in improving the accuracy of our algorithm.

2.1.1 Deducing Case IDs for Unlabeled Event Logs

The process of workflow mining requires event logs which usually consists the case ID, activity ID and the timestamp of the event. Event logs are hence very invaluable source of information about the process. Similar to any other data, event logs suffer from the problem of inconsistent and missing data. In case of event logs all the information it consist is important, but the absence of the case ID leads to an uncertainty in deciding the sequence of the events that took place during a particular case. Such an event log is called as unlabeled log. There are numerous reasons to having unlabeled logs such as, extracting the data from varied sources, human error or a lack of central system to monitor the business process. This is the reason why a pre-processing step is required to assign case IDs to unlabeled logs before we start analyzing the event logs. This paper[1] suggests a novel way to tackle this particular problem by deducing the case identifier (DCI) for such incomplete logs. In addition to the logs, this method (DCI) uses the executed process model and heuristics information about the execution times of the event as input, which finally results in a set of labelled logs. The input required here can be easily obtained from the business entity by accessing the documentation of the process model and from domain experts. The overall approach can be summarized by the three inputs as

1. Building the decision tree to help decide the events in the unlabeled logs.

2. Creating the behavioral profile which describes the process model in terms of the relation between activities in the log.

3. Additional information such as Activity Heuristics in terms of properties of the log. Here the deduced execution time of an event in the range average execution time plus/minus the standard deviation set by the user(), Ranking score function which sets the degree of trust between different possible event logs generated for an unlabeled event and the node probability.

The algorithm is slower than the existing expectation maximization algorithm but is much more accurate than it. The experiment is ran using a synthetic log generated to validate the results of unlabeled events. Overall the proposed model is very efficient in deducing the case IDs and can be used in complex process models as well. This approach has a drawback that it cannot be used in case of cyclic model.

2.1.2 Sequence Partitioning for Process Mining with Unlabeled Event Logs

The event log specifying the details of the process flow within an organization consists of several sequence of entries. Each entries uniquely identifies a case with case id. Case id is related to a task id, which is the sequence of operations performed for a particular process instance. In practical each of the sequence is very long, consisting of several hundreds and thousands of events. In such a scenario if the case id is found to be missing, then determining the workflow of the model becomes impossible as there can are chances of many interleaved events as well. [12] address in finding a solution of workflow mining from an unlabeled sequence of event logs when the attribute case id gets missing. Given a sequence of events, it becomes difficult to look for unique patterns. For example given a sequence xxyyzxy, it can have several solutions with patterns. The solutions can be (xy^2, xyz) or (xyz, xy^2). This is 2 pattern solution that indicates that xy is present 2 times and xyz is present once. Similarly we can even coin a 3 pattern solution: (x^3, y^3, z) or (x^3, z, y^3) or (z, x^3, y^3) or (z, y^3, x^3) or (y^3, z, x^3) or (y^3, x^3, z). [12] addresses this problem for the first time and aims in finding out the minimal pattern solution when the case id gets missing by searching through the entire search space. Several researches have been done in this field. Like the Expectation Maximization approach where the Markov model was used to determine the case id. It used a greedy approach to assign a case id to the event logs. But as suggested, since it is a greedy approach, it approaches towards result using the local maxima and fails to consider all the possible cases. And the Markov's model aims to assign the case if to the missing event log based on its own model, and looking for the local maxima, it fails to look for the entire search space. Thus this new approach leverages the concept of 'minimum description length (MDL)' [12]. Here the method suggested aims in solving the NP hard problem of interpreting the sequence of symbols in three steps. It uses the general sequencing concept to solve the problem. The steps are following:

1. In the initial stage, using complete patterns, a trie is built. These patterns occur in the input sequence and it also keeps a count of the occurrence of each of each of the symbol in the patterns.

2. The information stored in the trie helps to identify the candidate solutions by solving equations of systems. Each candidate stores the occurrences of the symbol coming up in the pattern.

3. Candidate solution is checked, if they are minimal solutions containing the entire search space.

This model was implemented across various case studies and proved to be efficient in finding the solution.

2.2 Traditional Workflow Mining

The paper by Aalst et al.[11] proposes by a solution to the workflow rediscovery problem. On the basis of complete workflow logs, a mining algorithm was to be found which can extract a big class of sound WF-nets. The alpha algorithm which does this thing is presented. When the paper was published, the authors were still trying to improve the algorithm. The improvement was that the algorithm should be able to find an even bigger class of WF-nets. The

problem of short loops was tackled and now research was being made to find duplicate and hidden tasks along with advanced routing constructs. However, the authors state that the rediscovery problem was not the only goal of their paper. Analysis of a workflow log without the understanding of any underlying knowledge and in the presence of outliers and noise was the ultimate goal of the paper. Their mining techniques were used to mine two real-world applications. The two applications were: 1) Health-care application and 2) CJIB (Centraal Justitieel Incasso Bureau). In the health-care application, analysis of patients belonging to multiple disciplines was made. One example of patients belonging to multiple disciplines was the logs of patients from Elizabeth Hospital in Tilburg. The patients were suffering from peripheral arterial vascular diseases. In regard to the second example, 130,136 cases were mined to process the information. 99 tasks comprised the process and the CJIB validated it. Thus, the empirical results demonstrates the efficiency of the alpha algorithm.

The paper by Buffett et al.[3] proposes a solution containing sequence classification which is used in control flow in municipalities for the identification of any statistically important differences. If the sequential pattern mining is value-based, then it can also be used in the identification of control flow which is linked to high or low throughput times and earlier or later process instances. For bringing the social-network related features of the data, the authors have used conventional process mining methods. The municipalities mentioned here are in Netherlands. The work is presented in stepwise format. In the first step, Common Lisp program was used to convert CSV file data into XES log files. During this activity, event activity codes were converted into English descriptions to enhance the readability. 5,647 cases and 262,628 events were handled successfully. The municipalities' respective logs contained only the frequently starting, middle and ending events. 80 percent selection threshold criteria was fixed in the ProM's 'Filter log using Simple Heuristics' module. Using the ProM's 'Visual Inductive Miner' module, process models were generated for each of the five municipalities. In doing so, only 90 percent of the most frequent sequences were used. Resource-activity matrices were made from the activities found in the process models. A task metric similar to one used previously was performed for the analysis of social networks using coefficients of resource degree correlation. Fruchterman-Reingold force-directed algorithm was used because it helped in the visual inspection of the social networks. Identification of clusters of resources operating on the same tasks was allowed by this visual inspection. Lastly, role clusters were used for the creation of resource-activity matrices for every municipality. The empirical results show interesting patterns hidden in the data. These were visualized using visual inspection.

The authors of this paper[5] described a new technique to construct process models. Their approach surpasses the limitations existing in the present methods, by being able to handle loops, duplicate tasks, long distance dependencies, and non-free choice constructs. They consider the whole sequence of events and construct patterns using Maximal Pattern Mining (MPM). Out of all the available methods so far, only DT Genetic Miner considers all the limitations such as, noise, duplicate tasks, hidden tasks, and loops. However, this approach takes more time to build a model, and renders low efficiency. MPM technique achieves all these and takes less time to mine and build model, along with better precision, recall and f-measure than other models. Unlike the variants of the alpha-algorithm, which consider only the relationship between two events, this method evaluates all events present in the trace and finds the optimal set of regular expressions that would cover them. The input to the algorithm is the collection of all traces present in an event log, such that it is ordered based on the number of events in a trace and within each trace, it is ordered based on the value of events. The event log is assumed to be having information regarding the event type/value, agent who is doing the event, the client who initiates the event, timestamp and data element being modified. Given the input, the model creates a pattern list based on which a graph is plotted. The first half of the algorithm contains five main components- finding self/sequence loops, storing frequent patterns and events in vertical format, identify events which occur in parallel, scrutinize if a trace is covered in the pattern, and pruning non-maximal patterns. Based on the loop property, the patterns are iterated through till all the loops have been discovered. The patterns are stored as an IdList in bitset representation, where each entry is an element, the id of the trace in which it appears and also the position where it appears. This step requires only a single scan through the logs. Also, in order to handle noisy data this keeps track of only frequent events (the ones which occur over a threshold value). The threshold value is based on trial and error. Once these are all set, for each event among the set of frequent events obtained, they are appended to a sequential pattern to form new patterns. A pattern is said to be maximal only if there is no other pattern that has the same start and accept states. After all the patterns are produced, the maximal patterns are checked for among these and then a final graph is plotted. The authors used synthetic and real event log data to evaluate their method. When synthetic data was used, the recall, precision and f-measure of this algorithm was higher than alpha variants algorithm, heuristic mining, and AGNES. The model showed similar performance when performed on real data as well. This algorithm produces more accurate results as it gets trained on more unique data. It can also handle unobserved behaviour in events up to an extent. One drawback in this is that it cannot handle duplicate tasks in parallel processes.

2.3 Workflow Mining with Interleaved Event Logs

The paper by Lou et al.[8] proposes an algorithm which is used for mining program workflows from interleaved traces. During system execution, these traces are used to record system events. The authors claim that their algorithm can be used to construct parallel workflows, unlike other algorithms. They have developed a three-step approach towards constructing the workflows. In the first step, for every pair of events, the temporal dependencies are mined. Here, only the valid dependencies are considered while the rest are ignored. In the second step, a basic workflow model is constructed which is based on the dependencies using a breadth-first path pruning algorithm. A preliminary workflow model which is very basic in nature is designed before the construction of the workflow from the traces. The dependencies are used for the construction of the preliminary workflow model. In the third step, verification of the workflow is done by re-

fining it with all the traces. Minimum thread types can be used for finding the simplest workflow by the refinement algorithm. The authors claim that their algorithm is very efficient based on the analysis of simulation and real program data. How the temporal dependency is mined is given in depth. To begin with, the statistical metrics of support and confidence is calculated. Then, the importance of dependency which is temporal and which takes place between two events is measured. These are the common steps with all the other sequence pattern mining algorithms. The confidence values and the support numbers can be obtained for the dependencies for every pair of the event by scanning the event traces. The running time complexity of the algorithm is O(Nd). Here, N stands for the total length of the event traces and d stands for the number of different event types. Unlike other algorithms, there is no chance of any irrelevant patterns developing and all the constructed patterns are relevant in terms of the software system's execution behavior. Then, a preliminary workflow model is constructed. This model may not have some of the workflow structures. So, the model is refined by which these missing structures may be recovered. Open source programs such as Hadoop and JBoss were used for the purpose of empirical evaluation.

One of the common tasks in process mining is to generate a workflow model using the knowledge of event logs, but the event logs may map to more than one workflow model.[6] Such a scenario can be seen in many web based applications and such logs are called as interleaved logs. The event log contains the following information; session id, timestamp id, user id and operation id. The solution proposed here takes in this data and extracts the workflow models from it. There are a few assumptions that it follows in doing so, such as; multiple operations in a session can map to the same or two different workflows and that the number of workflows are known. Experiments carried out to test the effectiveness of the proposed system uses synthetic data generated to simulate reality. The data for the experiments is generated a list of operations is created which is needed to extract the workflows in the process. This step is required to deduce the session ids from the event logs. The probability of an operation to appear in a workflow differs from one another implying that an operation is used more frequently than another. This probability is used in deriving the workflows from the logs and used as a basis of comparison. To generate session ids from the extracted workflows, the logs are specifically made to have the interleaved nature by randomly assigning operations from more than one workflow for a session. How this method works is by choosing the workflows based on the probability of the workflow in the log. The operations in the workflow are then selected by looking at their probability to occur in a given workflow. Combining these outputs the session ids are derived. The limitation of such approach can be pointed to having missing information or lack of domain knowledge to figure out the number of workflows present in the logs. The earlier can be tackled by the approach we found in the solution to missing case id problem.

2.4 Probabilistic Workflow Mining

Workflow mining refers to the process mining in any enterprise model. It is crucial for any and every organization. Process mining refers to the learning of the workflow models within an organization from the event logs. [7], proposes a new model for workflow model that address 2 major limitations: (i) Entire event log should not be assumed to be belonging of a single workflow model. As complex business structure like amazon, consists of multiple workflow models interleaved together. For example it includes the workflow models of shipping, return, purchase etc. (ii) the pattern of user's usage of the workflow model is not considered in the existing mining methodologies. But this is crucial in the current business models, as this supports the use of workflow models by multiple users at the same time. And a specific group of users are responsible to work with a specific workflow model only. Making the pattern of the user's usage a significant parameter for workflow mining. The authors leveraged these 2 features and worked upon a co clustering algorithms and improve the mining technology by clustering the operation sequence into appropriate clusters. The dataset to work with was the event log which consisted of the essential information about session id, workflow id, user id and the timestamp. This log provides the essential information about the user's activity and the series of operations performed by the user in a given session. Using this input to the co clustering algorithm 2 clusters were obtained: User Behavior Pattern cluster and the sequence cluster. The former cluster groups the different users with the similar usage behavior while the latter clusters the similar sequences and looks for the workflow model. Incorporating the user's behavior in co clustering helps in the improvement of the mechanism by discovering the relevant information about user's interacting pattern with the workflow. The model proposed was built in 3 steps:

1. A PST (Probabilistic Suffix Tree) was constructed using DFS mechanism to represent a particular user's access to the sequence of operations in a particular session.

2. Using prediction possibility, similarity is computed between a given sequence and PST. Following a distance matrix is formulated to store the distance between user and sequence pair. This was names as the 'user-buy-sequence' [7], distance matrix.

3. In the final step non negative matrix factorization was applied to co cluster the users defined by rows and sequences defined by columns.

The experimentation of this model was carried on synthetic dataset base on a 'web-based marketplace system' [7]. The data consisted of 4 different kinds of users and several workflows with unique set of operations. As a result to the experiment it was observed that the confusion matrix obtained consisted of almost perfectly assigned users to the respective groups. This proved that the co-clustering algorithm that worked on the basis of k-Means clustering outperformed existing models, evolving both the user's behavior cluster and sequence cluster appropriately. This new proposed model proved to be an effective mining method incorporating user's behavior pattern in analysis.

The authors of the paper[10] described a probabilistic workflow model whose learning algorithm runs in polynomial time. It is important to know the causal and probabilistic dependencies among the tasks to evaluate costs, design production policies, and to predict the effect of new policies. The paper discusses a probabilistic model for workflow graphs, and algorithms for learning those graphs from

data. The paper deals only with acyclic graphs. So the likelihood of a task being executed is dependant only on its parents or the direct pre-requisites. The authors use a AND/OR workflow graph. This essentially requires that each node should be present in the following classes: split node (node with multiple children), join node (node with multiple parents) and simple node (having only one parent and one child). The split nodes represent the point of choices and joint nodes represent the points of synchronization i.e, this node ensures that all threads which have the joint node as its end point to be completed before executing any of its own children. Another important step to be followed is to mark the events as happened or not happened, i.e. T=1 means that the event T happened. Using these a parametric model for Direct Acyclic Graph, determined by the conditional probability of each node is built. Another random variable defined is $P_{\alpha T}$ which is the representation of the joint state of parents of task T i.e., $P_{\alpha T} = 0$ represents the event where parents of T are assigned the value 0. So, P(T=1| $P_{\alpha T} = 0$) implies that a necessary condition for the event T to be executed is that at least one of its parents should be executed. The paper assumes that the data available is a workflow log. It contains records of which tasks were performed for which process instances and its starting time. In order to distinguish between two graphs, hidden variables are allowed. Non-simple nodes which cannot be measured can be defined as hidden variables. The model assumes that there is certain noise associated with measuring each task. The model also assumes Causal Markov condition, in order to calculate the effect of new policies that can change the probabilistic distribution of certain tasks. For the graph, an ordering oracle also needs to be generated. With these oracles, as the input to their algorithm, the workflow structure is determined. The algorithm keeps on adding child nodes, to partially built graph in a specific order. The paper uses a theoretical workflow followed by a company in Japan. From the analysis of e-mail logs, reports and summaries of documents, a graph was built manually. Applying the algorithm on this, produced mediocre results. So the authors wanted to incorporate Bayesian approaches in their future work. Even though this is the first approach to use coherent probability model, the algorithm still needs to be improvised as dynamically generated model are not statistically efficient as a constrained model.

2.5 Care-Flow Mining

The paper by Caron et al.[4] proposes a unique way of stimulating the progress of new perspectives. These perspectives are related to care-flow mining. Heterogeneity and multi-disciplinary nature characterize the human-centric process of health-care processes. The real challenge lies in discovering the process because each patient's treatment is unique. The authors have proposed new applications which result in getting useful information. This useful information belongs to care-flow mining techniques. They have also developed two focuses, one on department-based sub processes and the other on a specific treatment or drug. Handling claims at an insurance company or taking decisions about social security is an example of human-centric business process. The authors have used the event log from a Dutch hospital. They did a preliminary analysis of the log and found the assumption that the process is unstructured is true. When they used the heuristics miner plug-in, they obtained a spaghetti-like process model. They found there were hardly any repeating patterns. This was proved by the performance sequence diagram analysis tool. For the purpose of the analysis of the development-based sub processes, they used the example of radiotherapy department. For the purpose of the analysis of the use of a specific therapy, they investigated Paclitaxel, which is a cancer curing drug. When they analyzed the department-based sub processes, they made 4 observations. The first model was regarding the process model discovery with heuristics miner. They performed pre-processing of the log data before generating a process graph. Initially, the events which originated at the Radiotherapy department were captured by an event filter. They removed all the events of the activity 'administratief tarief eerste pol'. This was done because all such belong to administrative purposes and have no role in the health-care procedures of the department. They also added a virtual start and end event for each instance of the process. They also rationalized the radiotherapy options. The second observation was regarding the structure of the Dutch health-care system. The structure is layered. They made a distinction between the second and the third order medical care. A specialist at the polyclinic is contacted first by simulated patients. They removed the process instances that start at the polyclinic's first contact. They explained some of the reasons behind this removal. The patients may have first contacted an administrative staff or the time log might have started after the first event has taken place. In the third observation, the authors have discovered relationships with other sub processes in the care-flow. The sub processes in the log data were interlinked. They used the social networks miner plug-in with the handover metric. There exists some kind of relationship between the anesthesiology clinic, function center ENT, day center ward, medical microbiology, general lab clinical chemistry, internal specialism clinic, pathology, radiology, maternity ward, nuclear medicine, nursing ward, Obstetrics and gynecology clinic and recovery room/high care. They used the performance sequence diagram analysis plug-in to prove that there is no standard pattern of interaction. A frequency of 1 is achieved in each pattern where radiotherapy occurs. The fourth observation unearths a common sequence between treatments. In the department of Radiotherapy, the first treatment is the first teletherapy, the second is hyperthermia therapy and the third is Brachytherapy. This sequence is suggested by the heuristics miner. They have studied this phenomenon with the help of LTL-checker plug-in. They concluded that most of the cases in the log begin with teletherapy. For such cases, if they contain brachytherapy, then at least one session of teletherapy is used. Between teletherapy and brachytherapy, a session of hyperthermia is performed. When they analyzed the use of specific therapies, they again made 4 observations. While performing this analysis, the focus was on the use of Paclitaxel. In this analysis of process, the patient is the business object. They used Paclitaxel because of its high number of executions. The first observation was regarding AC-Paclitaxel chemotherapy. The authors failed to find any instance of patients who received both AC and Paclitaxel therapy. They proved that Paclitaxel was indeed used as a cancer-curing drug. The second observation was regarding the deviation between the actual average and the prescribed average number of cycles. They determined the approximate average of 7 treatment cycles and inferred that 4 cycles or

less was enough for 30 percent of the patients. They conclude that such analysis is impossible to perform without having exact timestamps. The third observation was about nurses administering paclitaxel. The authors concluded that a medical doctor is not needed to administer this drug and that most often, nurses administer it. The fourth observation was regarding the versatile use of paclitaxel. Along with cancer, the log data shows that paclitaxel is also used to cure uterus, endometrium and cervix malignancies.

2.6 Application of Process Mining in Healthcare - A Case Study in a Dutch Hospital

Below are the review of papers read for getting better understanding on the workflow model of the Dutch Hospital dataset. Our understanding has been enumerated in two below sections.

2.6.1 Flow identification of the Dutch Hospital Dataset

Just like any other business entity, a Hospital tries to maximize its benefits in the competitive world. Similar to a business process, a hospital has its own process of treatment, diagnosis, etc. To improve upon these processes, it is important to have a clear idea about the care flows the hospital follows. Identifying these care flows is a significant task as we have already looked at the heterogeneity of this event log. This paper[9] specifically helps in identifying the control flow and organizational perspective in order to provide new insights in a healthcare business process.

The paper explains the different process mining methods that are applicable to such a healthcare process due to its wide domain of application. Since, the healthcare data here contains an event log it is very much possible to extract the workflow from it similar to any other business process. The idea behind business process mining is to improve the underlying process, which involves discovering and monitoring of such processes. The control flow aspect of process modeling is not the only perspective a process can be worked on. Such an aspect allows us to discover the process model. Other aspects such as data flow or organizational perspective helps in conforming of the process model to the one designed by the business entity or improving a process model to benefit the same.

The study explains the process mining method that is applicable to our data by showing the possible interactions between control flows and organizational data. The study elaborates on the three different perspectives in process mining, which are, control flow perspective, organizational perspective and performance perspective. These techniques helped us evaluate the correct process mining task to follow, the selection of the perspectives in our logs and improving the process model.

2.6.2 Analyzing treatment procedures for patients

The dataset used for this project comes from a Dutch Academic Hospital and was used in the BPIC challenge 2011. The most difficult task in our study was the pre-processing of this event log. Since, this data is in a foreign language and comes from a real life process of treatment and diagnosis of patients it is very difficult to comprehend its heterogeneity. To explain the varied nature of this dataset it is enough to say that, one of the most powerful tools of workflow mining, ProM was unable to handle the number of events and traces it has. To generate the required workflow we first studied the data using the ProM plugin to filter events based on event and trace attributes of the diagnosis and treatment. Using these handy plugins we tried to understand the event logs. We are able to produce a number of interesting findings with the help of this pre-processing step. The study by Bose et al.[2] in the BPIC challenge also helps us understand the different perspective present in the event log.

This log contains event related to patients administered at the hospital, diagnosed with the cancer of the cervix, vulva, uterus and ovary. The log consists of four main perspectives in the data. They can be identified as

a. Diagnosis perspective: Most of the cases consists information about a persons illness by specifying it as a attribute which can be classified into diagnosis code and diagnosis. Both these attributes take on various values depending on the various regions it affect. There is clear combination of diagnosis code which tells us about relation between various diagnosis.[2]

b. Treatment perspective: Similar to diagnosis each case has a treatment code and treatment attribute associated with, which shows the combination of treatments that a patient went through. Though there isn't any description of these attributes.

c. Time perspective: The time perspective contains the date and time at which a particular case was associated with. Though this time perspective is different than the time stamp of the event logs.

d. Organizational perspective: For each event there exists an attribute which signifies the department that event took place at. There exist 43 such known departments throughout the logs. Some of the important ones present in most of the cases were General Lab Clinical Chemistry, Pathology and Radiology. The homogeneity of the data can be derived by working on one of these groups instead of looking at all.

e. Urgent and Non Urgent: Each activity in the log is manifested as urgent or non-urgent. The logs can be further grouped based on these values. For a specialized group we can subdivide it further based on any activity under it classified as urgent.

Generating a workflow for the whole log is incomprehensible hence we find homogeneous set of logs by using one or combination of the above perspectives which can be easily used to generate workflows from. After careful understanding of the data, we proceeded with filtering the logs initially on the diagnosis code combination of M16 and 821, which corresponds to the cancer of the ovary.

3. APPROACH TO SOLVE THE PROBLEM

After doing a considerable survey in the field of workflow mining we intend to help in decision making in the business process. It is quite essential in an organization to effectively take decisions and be process aware. For the same effective mining of the workflow model from the event log is crucial. For our hospital dataset, we intend to explore the workflow of departments when handling cancer patients and facilitate in making decisions.

We used ProM to generate the petri net model. ProM is a process mining tool which can be used to discover and analyze the process models from the event logs through its numerous tools and plug-ins. It provides various process mining algorithms like the alpha algorithm. It is very easy to install on any machine and very straightforward to use.

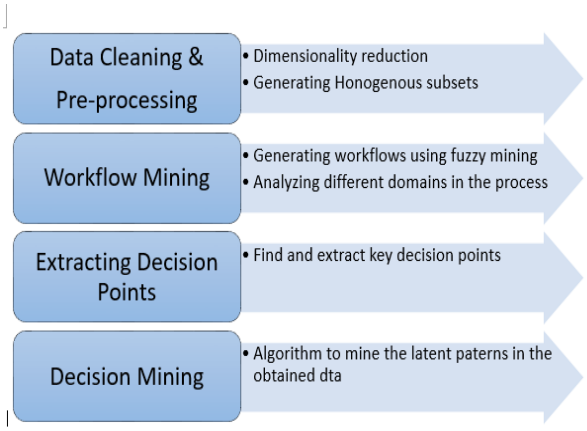

Figure 1: Flow of implementation

However, we must be aware of all the process mining algorithms available in ProM so as to make best use of it. Figure 1 below is the flowchart of our approach.

3.1 Data pre-processing

For our research, we wanted to figure out how the hospital handles the ovary cancer patients. As per the dataset, diagnosis codes M16 and 821 correspond to Ovary cancer treatment. We filtered the M16 and 821 cases from the input data. In Prom tool, we loaded our input log file (.xes format), i.e, the hospital dataset and using 'Filter Log on Trace Attribute Values' we filtered the M16 and 821 cases. After this filter we obtained 453 traces corresponding to 453 patients and 15394 events for all those ovary cancer related patients. On this filtered file, we used the organisation perspective to further pre-process the data i.e. to remove consecutive events belonging to the same organisation name (i.e. org:group). Java code has been written to do this processing.

We also renamed attribute 'org:group' to 'concept:name'. The 'concept' extension in xes format is a global level trace attribute based on which the process mining algorithms generate the workflow. Since we wanted to know the relations between department that is the org:group we replaced 'org:group' with 'concept:name'.

Now we have final dataset ready to be mined to look for the workflow of the process in it. Following we used 'Mine for a petri net using Alpha algorithm' and 'Fuzzy Miner' algorithm in the Prom tool were used.

1. **Alpha Miner:**Alpha Algorithm is a process discovery technique to discover workflow from the logs.

   The alpha algorithm designs the workflow net based on relations between event transitions present in the log. The set of all relations in a log is called footprint of log. Some of the relations defined are:

   (a) Direct succession (x > y) : This implies y occurs after x. Present in the log as xy.

   (b) Causality(x -> y) : This implies that x will be followed by y and y can never occur before x. Present in the log as xy and no occurrences of yx.

   (c) Parallel(x||y) : This implies x and y occur together. This follows a symmetric relation (a||b -> b||a) and no order is imposed on them. So both xy and yx are present in the logs.

   (d) Unrelated(x # y) : This implies that there is no particular order followed by either of the events. So this means that none of the traces are like xy or yx are present in the log.

   The alpha algorithm is built on the relations defined above.

   The steps include:

   (a) Form a set T which contains all the transition names from the log L.

   (b) Divide the set into T1 and T0; T1 containing all the initial transitions and T0 being the set having all end transitions.

   (c) Find pairs (A,B) such that tA $\in$ A should be connected to all tB $\in$ B via some place p $\forall$ a $\in$ A and $\forall$ b $\in$ B holds that a$\rightarrow$b $\forall$ a1,a2 $\in$ A : a1 # a2 and $\forall$ b1,b2 $\in$ B : b1 # b2

   (d) We select only the maximal sets among all the sets formed. (A maximal set is a set which contains the maximum number of elements that can be connected via a single place).

   (e) For each pair (A, B), all the elements in A are connected to all elements in B at one single place p(A,B).

   (f) Then we connect appropriate transitions with input and output places.

   (g) As the final step, we connect start node i to all transitions from T1, and all transitions from T0 to end state o.

   There are certain limitations to this algorithm:

   (a) It fails to re-discover short loops of length one and two.

   (b) It doesn't consider some process constraints due to which non-local dependencies arise. This is a major drawback in many of the process mining algorithms.

   (c) We have used this algorithm to find our workflow model for the Dutch hospital set. Since the output of this is difficult to visualise, we also used Fuzzy miner algorithm to get an idea about the workflow model.

2. **Fuzzy Miner:** This algorithm is used to visualize an unstructured process flow. In real life scenarios, the logs are not well structured. When mining algorithm are applied on these unstructured logs, it tends to generate spaghetti like process flow, not getting us to get an insight to the actual process flow. Though the models generated from the mining algorithm like alpha miner may be difficult to visualize, fuzzy miner makes this process much easy by providing a simplified on the process of the flow.

   Fuzzy miner uses the roadmap metaphor to create the visualization of the process. This algorithm performs in 4 different steps which are as follows:

(a) **Log-based Metrics:** In this step it analyzes the entire log and evaluates the relation between different events and the importance of each of them in the process. The importance of the events in the process is evaluated using 2 concepts: (i) significance and (ii) correlation. Significance helps in determining the importance of the event by measuring the same for the activities involved and the relationships present over the events. On the other hand correlation is measured only for the two events for its precedence relation. It measures how close the relation is between events.

(b) **Initial Model Generation:** Using the log based metrics as computed in the previous step, all the event classes are determined from the log and transformed to the activity nodes. In the process model, a directed edge is added for every precedence relation existing for the event.

(c) **Adaptive Simplification Process:** In many a times, the model obtained after the second step is quite complex. And thus certain simplification algorithm is used to make the model look simple. To name a simplification algorithm in this regard is graph simplification algorithm. It uses the different parameters of fuzzy miner like NodeCutoff, EdgeCutoff, UtilityR, Preserve and Ratio to design the simple graph model of the process.

(d) **Model Construction and Evaluation:** After the model has been constructed, the algorithm makes it ready for the visualization.

The workflow model was generated using alpha miner and saved the output in PNML format. PNML (Petri Net Mark-up Language) was essential for the continuing our final analysis of the data. Since the workflow generated with the apha miner algorithm was too complex to understand, we applied Fuzzy miner ('Mine for a Fuzzy Model') on the input file, to generate better visualisation of the workflow model. The output Fuzzy model, gives us the flow and decision points visually. PNML output given by the Alpha algorithm was used to obtain the decision points. A decision point is a node which has more than one outgoing arc. We wrote a java code findDecisionPoints.java to find the decision points in the workflow. The algorithm used is[13]:

Input: PNML file generated by the alpha miner

Output: List of decision points

(a) Initialize a Map map1 to store the transition_id and event name.

(b) Initialize a Map map2 to store the transition_id and its frequency.

(c) for each transition in P
Add entry<transition_id,event_name> to map1

(d) for each arc in P if (source is in map2) Increment value in map2 where key == source

(e) for each key in map2
if (map[key].value > 1)
make the key a decision point

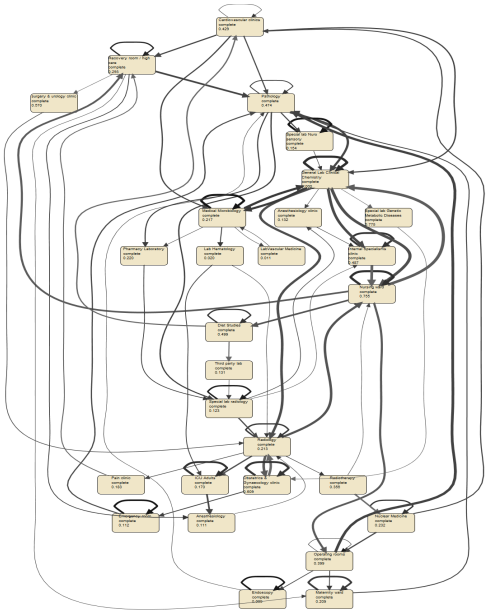

Figure 2: Workflow Model of Ovary Cancer Patients

In the next step, we found the different sub-sequences each decision point can form with a given window size and extracted the data for that decision point. The concept of window is used in extracting the data. The window slides along the event trace, and forms subsequences for a given decision point, where for a window size of i, all the points upto the decision point are considered, if the decision point occurs at ith index. So for a workflow with m decision points i.e., nodes[n1,n2,n3 ... nm] : If the decision point is node nk and the window size is i then the subsequence formed is [nk-i, nk-(i-1), nk-(i-2), ..., nk].While extracting the data from the dataset, we always considered the trace attributes, and the event attributes of the last event occurring in the queue. In the log file we have identical trace and event attributes for all the traces.

For our implementation we considered a window size of 3. For every subsequence we extracted the trace attributes and event attributes along with the target variable i.e., next event in the trace. This was done in order to generate the classification model that would predict which is the next event performed by a patient when she reached a decision point.

The trace attributes we considered are:

Treatment code, Treatment code:1, Treatment code:2, Treatment code:3, Treatment code:4, Diagnosis code, Diagnosis code:1, Diagnosis code:2, Diagnosis code:3, Diagnosis code:4, Specialism code, Specialism code:1, Specialism code:2, Specialism code:3, Specialism code:4, Diagnosis Treatment Combination ID, Diagnosis Treatment Combination ID:1, Diagnosis Treatment Combination ID:2, Diagnosis Treatment Combination ID:3, Diagnosis Treatment Combination ID:4, Age, Age:1, Age:2, Age:3, Age:4.

The event attributes we considered are: Specialism code, category:name, Number of executions,Producer

code, Section, Activity code.

We wrote java code findSubSequencs.java to do the same. The algorithm used:

Input: the filtered XES log file and list of decision point

Output: Separate files with extracted data corresponding to every decision point.

(a) Initialize a Map map1 with key as subsequence and value as occurrence.

(b) Initialize a List list1 to store the attribute values.

(c) Initialize a queue Q of size of the window.

for each of the decision point

for each event in trace

Add event to the queue.

if (queue is full and the event is the decision point)

Write the list to the file.

if (key is in map1)

Increment value that indicates 'occurrence'.

Write the list to the file.

Reset the queue

Now we have obtained the data in a file with all the attributes and its value. Our dataset contains the input variables and target variables. For developing the classification model, the data needs to undergo further pre-processing so that we can generate accurate classification model. In this phase we considered the data for the decision point corresponding to 'General Lab Clinical Chemistry'. This was because this department had the maximum number of data points associated with it. And we wanted to make the decision easy by predicting, which is the next step in the process flow to be followed after reaching this decision point. In preprocessing step we removed the attribute: "Number of executions" and "Section" as they contained the same value for all the records and it would not have any contribution towards making a decision. The missing attributes were replaced with 0. After this the preprocessed data file was passed in Weka and worked with J48 (Decision Tree). The model in figure 3 could only provide an accuracy of 60 percent only.

4. WORK IN PROGRESS

1. After generating the decision tree, we are running various machine learning algorithms to develop a good decision making model. The approaches that we are considering are:- J48, Random Forest, Logistic Regression, kNN and Naive-Bayes.

2. Following this, we would generate Receiver-Operator-Characteristic curve to evaluate and compare each of these classification models. ROC curve plots the true positive rate with the false positive rate for a classification model.

3. The ROC curve which gives the maximum true positive and minimum false positive will determine the best classification model.

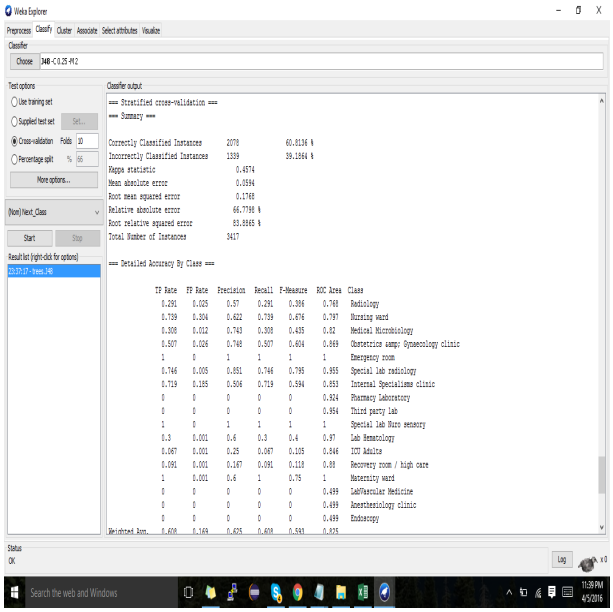

Figure 3: J48_output

4. Now, we have only considered 'General Lab Chemistry' as our decision point and building the classification models but the work can also be extended for other decision points obtained from the workflow model.

5. REFERENCES

[1] D. Bayomie, I. M. Helal, A. Awad, E. Ezat, and A. ElBastawissi. Deducing case ids for unlabeled event logs. *algorithms*, 3(3.2):3–2, 2015.

[2] R. Bose and v. d. W. Aalst. Analysis of patient treatment procedures: The bpi challenge case study. 2011.

[3] S. Buffett and B. Emond. Using sequential pattern mining and social network analysis to identify similarities, differences and evolving behaviour in event logs.

[4] F. Caron, J. Vanthienen, J. De Weerdt, and B. Baesens. Beyond x-raying a care-flow: Adopting different focuses on care-flow mining. *BPI Challenge*, 2011.

[5] V. Liesaputra, S. Yongchareon, and S. Chaisiri. Efficient process model discovery using maximal pattern mining. In *Business Process Management*, pages 441–456. Springer, 2015.

[6] X. Liu. Unraveling and learning workflow models from interleaved event logs. In *Web Services (ICWS), 2014 IEEE International Conference on*, pages 193–200. IEEE, 2014.

[7] X. Liu, H. Liu, and C. Ding. Incorporating user behavior patterns to discover workflow models from event logs. In *Web Services (ICWS), 2013 IEEE 20th International Conference on*, pages 171–178. IEEE, 2013.

[8] J.-G. Lou, Q. Fu, S. Yang, J. Li, and B. Wu. Mining program workflow from interleaved traces. In *16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 613–622, July 2010.

[9] R. Mans, M. Schonenberg, M. Song, W. M. van der Aalst, and P. J. Bakker. Application of process mining in healthcare–a case study in a dutch hospital. In *Biomedical Engineering Systems and Technologies*, pages 425–438. Springer, 2008.

[10] R. Silva, J. Zhang, and J. G. Shanahan. Probabilistic workflow mining. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 275–284. ACM, 2005.

[11] W. van der Aalst, T. Weijters, and L. Maruster. Workflow mining: Discovering process models from event logs. In *IEEE Transactions on Knowledge and Data Engineering*, pages 1128–1142, September 2004.

[12] M. Walicki and D. R. Ferreira. Sequence partitioning for process mining with unlabeled event logs. *Data & Knowledge Engineering*, 70(10):821–841, 2011.

[13] J. Xia. Automatic determination of graph simplification parameter values for fuzzy miner. *Eindhoven University of Technology. Netherlands*, 2010.