Noname manuscript No.
(will be inserted by the editor)

Distributed Process Mining

Asjad Khan · Aditya Ghose

the date of receipt and acceptance should be inserted later

Abstract Process Mining can help healthcare organisations to streamline and improve patient treatment processes(also known as 'careflows'). However Privacy concerns may prevent the Healthcare organizations from directly sharing the data. Current process mining tools work by gathering all datasets into a central site, then running various algorithms against that data. In most practical cases privacy concerns can prevent building of such a centralized data warehouse. Thus event- log data may be distributed among several custodians, none of which are allowed to transfer their data to another site. In this paper we present a novel method that extends the Flexible Heuristic Miner algorithm to address the problem of Secure Distributed Process Mining. The presented method allows us mine geographically dispersed process logs with minimum information sharing between the parties involved.

Keywords Process Mining · Distributed · Privacy

1 Introduction

Privacy concerns can prevent organisations from directly sharing the data(such as process logs). Current process mining tools operate by gathering all data into a central site, then running various algorithms against that data. However, in most cases privacy concerns can prevent us from building a centralised data warehouse. Thus event-log data may be distributed among several custodians, none of which are allowed to transfer their data to another site [1][2]. A classical example of a privacy- preserving process mining problem of the first type is from the field of medical research. Consider the case that a number of different hospitals wish to jointly mine their process logs for the purpose of medical

M. Asjad
University of Wollongong
Tel.: +61-45-1312633
E-mail: asjad@outlook.com

research. Furthermore, let us assume that privacy policy and law prevents these hospitals from ever pooling their data or revealing it to each other, due to the confidentiality of patient records. In such a case, classical data mining solutions cannot be used. Rather it is necessary to find a solution that enables the hospitals to compute the desired data mining algorithm on the union of their databases, without ever pooling or revealing their data. we will assume that the result of the process mining algorithm is either safe(in terms of privacy) or deemed essential[2]. Thus, the problem we address is how to compute the results without pooling the data, and in a way that reveals nothing but the final results of the process mining computation. Most typical methods presented in the literature rely on some form of data transformation in order preserve user privacy. These techniques are a trade-off between information loss and privacy[4][2].

In this work we extend the Flexible Heuristic Miner [10] to work in a distributed setting while adding privacy preserving components with an aim to minimize sharing of sensitive information between parties involved in the computation. In the Following sections an overview of process modeling and process mining is presented, followed by brief overview of applications in bussiness and healthcare. Finally with the help of an example we present our novel technique for mining geographically distributed process logs.

2 Background

Process Modeling and Process Mining

A business process is a collection of related and structured activities or tasks that focus on providing a specific service or product. Business Process modelling is used to represent information flow, decision logic and business activities in a business process. It can also be seen as ordering of work activities across time and place with clearly defined input and outputs representing a structure for action[10]. Process models are widely used in organisations to document procedures while assisting organisations in managing complexity and providing new insights[10]. Process mining is used to discover, monitor and improve real processes by extracting knowledge from event logs readily available in organizational information systems[10].

Process Mining lies at the cross section of data mining and business process modeling and and can be seen a process management technique that allows for the analysis of business processes based on eventlogs[10]. It is used to discover, monitor and improve real processes by extracting knowledge from event logs readily available in organizational information systems[10]. Process mining bridges the gap between traditional model based process analysis(e.g., simulation and other business process management techniq ues)and data centric analysis techniques such as machine learning and and data mining[10]. Process mining seeks to correlate event data(i.e observed behavior) and process models (hand-designed or discovered automatically using process logs). Events can take place anywhere. For example an enterprise information sys-

tem may produce an event log consisting of a various tasks completed and with the corresponding timestamp. Similarly, information systems or administrative systems deployed inside a healthcare orgnizations can produce an event log whose analysis can lead to discovery of process model being followed by that particular organization. Process mining strives to provide a concise assessment of processes being followed in reality, thus helping organizations in verifying, improving and redesigning process models. This is achieved by extracting knowledge and insights from event logs, which are generated by most of todays information systems. Typically, these event logs contain information about the start or completion of process steps, sometimes coupled together with related context data (e.g. actors and resources)[1][2].

The first application of process mining is process discovery where given an event log a process mining algorithm will produce a process model(explaining the behavior recorded in the log) without utilising any a-priori information[10][11]. The second type of process mining is conformance, used to check process conformance where we compare an existing process model with an event log of same process model. The idea here is to check if the actual process model(as recorded in the event log) aligns with the initially designed and deployed process model. The third type of process mining is enhancement, which is used to enhance or extend an existing process model by using information recorded in the event log[10]. For example by using timestamps and frequencies, we can identify bottlenecks, diagnose performance related problems and analyse throughput times. This helps us explore different process redesign and control strategies[11][16].

Privacy Preserving Distributed Process Mining

The field of Distributed Privacy Preserving Process Mining remains largely unexplored. Some related work has been done by Wil M. P. van der Aalst et. al in [25] to distribute process mining problems over a network of computers such as multicore systems, grids, and clouds etc. However significant research has been done in the related field of privacy preserving distributed data mining, which borrows knowledge from various computer science fields. Researchers from multiple communities such as the database community, the data mining community, the statistical disclosure control community and the cryptography community have explored this field independently[26]. The key goal of privacy preserving data mining is to mine useful models based on the entire distributed dataset without compromising the privacy of data in individual datasets[27]. Thus participants in this sort of computation may not fully trust each other in terms of sharing their individual datasets but maybe willing to collaborate in other ways.

Privacy Preserving distributed data mining(PPDM) problems can be categorized and classified based on following aspects[28]:

3 Secure Distributed Process Mining

Problem Definition

Let us assume that a transaction database DB(containing events logs) is horizontally partitioned among n sites (namely, $S_1, S2, ..., Sn$) where DB= $DB_1 U DB_2 U ... U DB_n and DB_i$ residesatsideSi (1<i<n). The databases contain large number of event logs generated by information systems deployed internally in healthcare organizations. The sites communicate by message passing. Our goal is to discover a global process model, satisfying the thresholds(as defined in). Furthermore, no site should be able to precisely learn the order and frequency of tasks occurring in the process log of other sites(i.e information disclosure should be minimum), unless that information is revealed by the knowledge of own dataset and final results. In this case we shall assume no collusion.

Method

The proposed Method follows the general approach of Flexible heuristic Miner (FHM)[22]. In this work, FHM will be extended to work in a distributed setting and privacy preserving protocols will be incorporated to limit sensitive information disclosure during broadcasts.

Running Example

The process model show in figure 1. was built using the data collected from an administrative system within a Dutch hospital(azM). Data is collected as part of the requirement by Dutch law where all hospitals need to record the diagnosis and treatment steps of individual patients to estimate medical costs[31].

The data however is a specific subset(from 2009 till 2012) representing the process followed for treating gastroenterology patients suffering from rectum cancer for which surgery is needed. Blue color circle in the model indicate a low waiting time. i.e less than 5 days waiting period. A yellow color circle is an indicator of wait time between 5 and 10 days, and lastly pink color circles indicate a waiting time of more than 10 days. The thickness of the arcs indicates the frequency(number of times path has been followed). Grey rectangles represent medical departments inside the hospital and the black rectangles are used for routing[31].

We will use this model to generate an artificial event log with 500 random traces. This will then be used to illustrate the different mining steps of FHM. We will start off by analyzing the process log for causal dependencies[22]. This type of relationship can be extracted from a dependency graph[21]. For the sake of completeness we will present here the definitions used in [21] and [22]. Process models built using FHM are represented by Causal Nets. Following definition [21]is used to define the concept of causal net.

Definition 1 *Causal net (C-net)*: A C-net is a tuple *(T,I,O)*, where

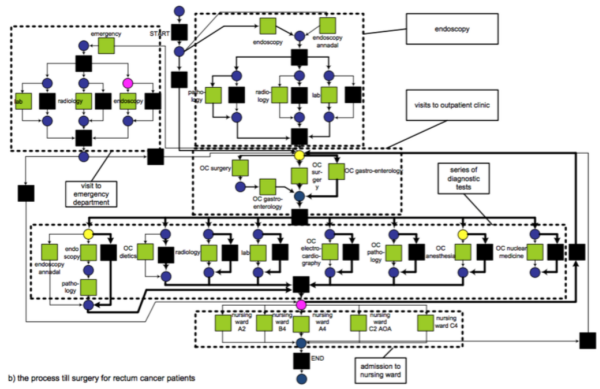

Fig. 1: Reference Process log for generating event logs

- T is a finite set of tasks,

- I: $T \rightarrow \mathcal{P}(\mathcal{P}(T))$ is the input pattern function

- O: $T \rightarrow \mathcal{P}(\mathcal{P}(T))$ is the output pattern function

If ϵ then $e \in T$ then $\Box e = \bigcup I(e)$ denotes the input tasks of e and $e\Box = \bigcup O(e)$ the output tasks of e.

Definition 2 *Dependency Graph (DG)*: If (T,I,O) is a Causal net then the corresponding Dependency Graph (DG) is a relation on T (DG $\subseteq$ T x T, with

- $DG = (a,b) \mid (a \in T \wedge b \in a\Box)\ (b \in T \wedge a \in \Box b)$

X	Activity	X
{ }	A	$\{B\}$, $\{C\}$
$\{A\}$	B	$\{E\}$
$\{A\}$	C	$\{D\}$
$\{C\}$	D	$\{F\}$
$\{B\}$	E	$\{F\}$
$\{D\}$, $\{E\}$	F	$\{G\}$, $\{H\}$
$\{F\}$	G	$\{I\}$
$\{F\}$	H	$\{I\}$
$\{G\}$, $\{H\}$	I	{}

Table 1: THE TRANSLATION OF PETRI NET (FIGURE 1) INTO A C-NET

Let us consider the Petri net in Fig. 1. It depicts several allowed paths for treatment. It can start with patient endoscopy, followed by diagnostics tests or other possibility is that patient visits outpatient clinic of gastro-enterology.

Task	Symbol
Visit Doctor	A
Visit Outpatient Clinic	B
Visit Outpatient Clinic	C
Diagnostic Tests(basic)	D
Diagnostic Tests(complete)	E
Consult Doctor	F
Visit Emergency Room	G
Admit Hospital	H
Hospital Discharge	I

Table 2: TASK SYMBOL MAPPING

This is followed by the patient visiting medical labs for a series of diagnostic tests. After diagnostic tests patient may consult the surgery doctor(either by phone or visiting the hospital. After the tests, there is a possibility that patient may visit the emergency department or admitted to the hospital[30][31]. From the description above we can conclude the careflow process has 9 Major tasks(where some of the tasks can be further subdivided). We can donate the tasks with symbols (A,B,....,I) so the corresponding task set T = A,B,....I . The following table represents the tasks and their representative symbols.

4 Privacy Preserving Distributed Flexible Heuristic Miner

Overview:

Flexible Heuristic miner is described in detail in [22]. In this work we will extend the FHM algorithm to work in a distributed setting. Generating the process model from distributed event logs will consist of two phases. In the first phase frequency matrixes are exchanges between all sites to compute a global dependency matrix. From this matrix we can generate a dependency graph. In the second phase the goal is to extract a C-net with dependency graph and event log as input. The details of these two phases and steps involved are given below.

Phase 1:

In this phase event log(as shown in fig.) of each site will be analyzed for casual dependencies. e.g if one task in the event log is always followed by another task it is likely that there exists a dependency relationship between both tasks[22]. The dependency relationship can be represented by using a dependency graph(DG). Following definitions [21] are later used to define the concept of DG.

Definition 3 *(Basic Relations)*: Let T be a set of tasks. $\delta \in T^*$ is a process trace, W: $T^* \rightarrow \mathcal{N}$ is a process log, and $a,b \in T$:

1) $a >$w b iff there is a trace. $\delta = t_1t_2t_2t_3...tn$ and $i \in \{1,...,$n-2$\}$ such that $\delta \in W$ and $t_i = t_{i+2} =$ a and $t_{i+1} = b$ (direct succesor),

2) a >>w b iff there is a trace $\delta = t_1t_2t_3t..t_n$ and $i \in$ {1,...,n-2 } such that $\delta \in W$ and $t_i = t_{i+2} = a$ and $t_{i+1} = b$ and $a \neq b$ (length-two loops),

3) a >>>w b iff there is a trace $\delta = t_1t_2t_3t...t_n$ and $i < j$ $i,j \in$ {1,...,n } such that $\delta \in W$ and $t_i = a$ and $t_j = b$ (direct or indirect successor),

Definition 4 *(Dependency Measures)*: Let W be an event log over T, $a,b \in T$, $|a >$ w $b|$the number of times $a >$w b occurs in W, and

$a \Rightarrow w\ b\ = (\frac{|a >wb|-|b >wa|}{|a >wb|+|b >wa|+1})$ if (a $\neq$ b) (1)

$a \Rightarrow w\ a\ = (\ \frac{|a >wa|}{|a >wa|+1})$ (2)

We take the following steps for the mining of the dependency graph(DG):

Step i: Exchange Task List

We start off by constructing set T. i.e a set containing all tasks appearing in all process logs of all the sites involved in the computation. Site 1 will pass the its set of task list T to site 2. Site 2 will adds its own task list and will pass it on to the third site. The process is repeated until we reach site 1 again. Site 1 will remove any duplicates found in the list resulting in set T.

Step ii: Compute Direct Successor Matrix

In step 1, site 1(designated as master site) computes a direct successor frequency matrix U that captures the dependency measures using the direct successor formula(given in equation 1) and task list generated from the previous step. The frequency metrics essentially indicate the certainty level of a dependency relationship between two events A and B(represented by A W B). The resulting values will be used later in a heuristic search or a threshold based approach in mining the and/XOR Splits.

	Start	A	B	C	D	E	F	G	H	I	End
Start	0	500	0	0	0	0	0	0	0	0	0
A	0	180	170	36	0	30	0	84	0	0	0
B	0	0	20	46	336	0	0	54	44	0	0
C	0	0	37	0	393	0	0	70	0	0	0
D	0	0	0	74	0	79	243	0	51	53	0
E	0	0	153	49	0	67	231	0	0	0	0
F	0	0	0	12	14	19	0	167	129	159	0
G	0	0	0	70	0	70	0	0	0	360	0
H	0	0	43	0	37	49	0	0	69	198	0
I	0	0	0	0	0	0	0	0	0	0	500

Table 3: DIRECT SUCCESSOR (a $\Rightarrow_w$ b COUNTING) Matrix

The resulting values of a W b will lie in a range of -1 and 1. A high value(closer to 1) indicates that we are quite certain that the dependency relation holds and vice versa.

Step iii: Generate Loop counting matrix

Site 1 further generates a length two loops counting matrix. Values of this matrix are computed by applying equation (3). Resulting matrix will contain frequencies of length two loops.

Step iv: Compute global DS Matrix using SMC

In this step we compute the global Direct Successor matrix and length n loop matrix. To ensure privacy, we shall employ the secure sum protocol proposed in [30]. This approach treats privacy preserving process mining as a special case of secure multi-party computation. In following this approach, we emphasize practicality and recognize that certain kind of information can be exchanged(e.g task list) without violating security policies. Strict following of SMC however forbids leakage of any information other than the final result. Our goal here is to jointly mine the event logs while keeping the individual inputs private. It should be noted that in this case, we assume a secure channel for communications exists and that three or more parties participating with no collusion.
We start off at the Master Site(site 1). Site 1 will generate a random number matrix R of length m x m(m = length of task list), where each element r of the matrix is chosen from [1..n]. In our chosen example, it will be a matrix of size 9 x 9 and each element will lie in the range of [1...500*k], where k is the total number of sites involved in the computation. Site 1 then adds this matrix to the direct successor frequency matrix(table 1) and sends the sum R + U mod n to site 2. As the values in the matrix R are chosen uniformly from [1...n] the values in the resulting sum matrix(R + U mod n) are also distributed uniformly across this region. Thus we site 2 learns nothing about the actual values of elements of matrix U[30]. This procedure is repeated by each site. Site l receives matrix V.

$$V = R + \sum_{j=1}^{l-1} v_j mod n$$

As values are uniformly distributed across [1..n], site l learns nothing about the original contents of the matrix. Site l then computes

$$R + \sum_{j=1}^{l-1} v_j mod n = (v_j + V) mod n$$

And passes it to site l + 1
Site s performs the above calculation and sends the resulting matrix to site 1, which then subtracts the originally generated random matrix R thus giving us the global Direct Successor Matrix. Same Procedure is followed to calculate the global length two loop counting matrix.

Step v: Mining Dependency Graph: We apply the DG-Algorithm on the resulting matrices to generate a dependency graph. But before this step, we sometimes add artificial tasks to indicate the starting and end point of the process. Usually this case occurs when clear unique start and end points dont exist in the given event log. Following definition [21] is used to define artificial starting and ending points.

Definition 5 *(Start/end extension)*: Let W be an event log over T. Then W^+ is the (aritificial) start/end-extension over T^+ with

1) $T^+ = T \text{ U } \{ \textit{ start,end } \}$
2) $W^+ = \{ \textit{start } \delta \textit{ end } | \delta \in W \}$

Definition 6 *(Dependency Graph (DG) - algorithm)*:

Let W be an event log over T, W^+ an event log over T^+ (i.e, the start/end-extension of W), σ_a the (absolute) Dependency Threshold(default 0.9), σ_{L1L} the Length-one-loops Threshold(default 0.9), σ_{L2L} the Length-two-loops Threshold(default 0.05), $DG(W^+)$ (i.e the dependency graph for W^+ *is defined as follows*

1) $T = \{ \text{ t } | \exists_{\sigma \in W^+} [t \in \sigma \text{] } \}$ (the set of tasks appearing in the log),
2) $C_1 = \{ \textit{ (a,a)} \in T \; X \; T \; | a \Rightarrow_w a \geqslant \sigma_{L1L} \text{ } \}$ (length-one loops)
3) $C_2 = \{ \textit{ (a,b)} \in T \; X \; T \; | \textit{(a,a)} \notin C_1 \wedge \textit{(b,b)} \notin C_1 \wedge a \Rightarrow_{2w} b \geqslant \sigma_{L2L} \text{ } \}$ (length-two loops)
4) $C_{out} = \{ \textit{ (a,b)} \in T \; X \; T \; | b \neq \textit{End} \wedge \text{a} \neq \text{b} \wedge \forall_{y \in T} [\text{ a} \Rightarrow_w \text{b} \geqslant \text{a} \Rightarrow_w \text{y }] \}$ (for each task, the strongest follower),
5) $C_{in} = \{ \textit{ (a,b)} \in T \; X \; T \; | b \neq \textit{Start} \wedge \text{a} \neq \text{b} \wedge \forall_{y \in T} [\text{a} \Rightarrow_w \text{b} \geqslant \text{x} \Rightarrow_w \text{b }] \}$ (for each task, the strongest cause),
6) $C'_{out} = \{ \textit{ (a,x)} \in C_{out} \; | (a \Rightarrow_w x) < \sigma_a \wedge \exists_{(b,y) \in C_{out}} [a,b \in C_2 \wedge ((\; b \Rightarrow_w y) \text{ - } (a \Rightarrow_w x) > \sigma_r) \text{] } \}$ (the weak outgoing-connections for a length-two loop),
7) $C_{out} = C_{out} \text{ - } C'_{out}$ (remove the weak connections),
8) $C'_{in} = \{ \textit{ (x,a)} \in C_{in} \; | (x \Rightarrow_w x) < \sigma_a \wedge \exists_{(y,b) \in C_{in}} [a,b \in C_2 \wedge ((\; y \Rightarrow_w b) \text{ - } (x \Rightarrow_w a) > \sigma_r) \text{] } \}$ (the weak outgoing-connections for a length-two loop),
9) $C_{out} = C_{in} \text{ - } C'_{in}$ (remove the weak connections)
10) $C''_{out} = \{ \textit{ (a,b)} \in T \; X \; T \; | a \Rightarrow_w b \geqslant \sigma_a \vee$
$\exists_{(a,c) \in C_{out}} [((\; a \Rightarrow_w c) \text{ - } (a \Rightarrow_w b) < \sigma_r) \text{] } \}$,
11) $C''_{in} = \{ \textit{ (b,a)} \in T \; X \; T \; | (b \Rightarrow_w a) \geqslant \sigma_a \vee$
$\exists_{(b,c) \in C_{in}} [((\; b \Rightarrow_w c) \text{ - } (b \Rightarrow_w a) < \sigma_r) \text{] } \}$,
12) $\text{DG} = C_1 \text{ U } C'_2 \text{ U } C''_{out} \text{ U } C''_{in}$

s

Output: Resulting Depdency graph is shown in table layout in Table 4.

Step vi: Pruning using thresholds.

In our approach we are going with frequency based metric approach where apply three kinds of thresholds. After computing the global dependency matrix we can apply a Threshold parameter to get rid of uncertain dependency relations, possibly caused by noise in the dataset. Following threshold parameters help in deciding if a dependency relation is incorporated in the final

X	Activity	X
{ }	A	$\{B\}$, $\{C\}$
$\{A\}$	B	$\{E\}$
$\{A\}$	C	$\{D\}$
$\{C\}$	D	$\{F\}$
$\{B\}$	E	$\{F\}$
$\{D\}$, $\{E\}$	F	$\{G\}$, $\{H\}$
$\{F\}$	G	$\{I\}$
$\{F\}$	H	$\{I\}$
$\{G\}$, $\{H\}$	I	{}

Table 4: Resulting DG in Table Layout

dependency graph or not.
1) Dependency threshold
2) length one-two threshold
3) length one threshold

Phase 2: In the previous Phase we had mined a dependency graph using the event logs from site 1...n. The dependency graph gives us information about the dependency between the tasks, but so far doesnt include any information about splits/joins [22]. In this phase well characterize split and join points of the DG. We do so by mining different split and join patterns for each task. E.g output set of task A is B,C. We need to figure out if there exists an AND-Split(i.e A is followed by both B and C) or an XOR-Split(i.e A is followed by B or C) between them. Mining of splits/join will require the dependency graph generated in the previous phase and the event log from each site. Output will be a augmented C-net containing bags rather than sets indicating the frequency of specific split and join patterns [22]. The concept of augmented C-net is defined in the following definition[21].

Alternatively, given a dependency graph, we can find out the frequency of input/output binding by replaying the event log on the DG. Various replay strategies exist

Definition 7 *(Augmented C-net)*: An augmented-C-net is a tuple *(T,I,O)*, where

- T is a finite set of tasks,

- *I:* $T \rightarrow \mathcal{P}(\mathcal{P}(T) \rightarrow \mathcal{N}$ *)* is the input frequency function

- *O:* $T \rightarrow \mathcal{P}(\mathcal{P}(T) \rightarrow \mathcal{N})$ is the output frequency function

Alternatively, given a dependency graph, we can find out the frequency of input/output binding by replaying the event log on the DG.

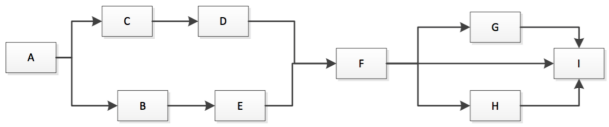

Fig. 2: Resulting Dependency graph (DG)

Output:

I	Activity	O
{ }	A	[$\{B^{360}\}$, $\{C^{340}\}$]
$[\{A^{336}\}]$	B	$[\{E^{336}\}]$
$[\{A^{336}\}]$	C	$[\{D^{309}\}]$
$[\{C^{336}\}]$	D	$[\{F^{213}\}]$
$[\{B^{336}\}]$	E	$[\{F^{227}\}]$
$[\{D^{360}\}, \{E^{340}\}]$	F	$[\{G^{360}\}, \{H^{340}\}]$
$[\{F^{157}\}]$	G	$[\{I^{157}\}]$
$[\{F^{157}\}]$	H	$[\{I^{157}\}]$
[$\{G\}$, $\{H\}$	I	{}

5 Analysis:

Correctness: The Proposed Privacy Preseving distributed Flexible Heuristics Miner (FHM) behaves in the same manner as the original FHM algorithm. i.e Given a process log P, and a set of process logs $P_1, P_2, ..., P_n, where P_1 U P_2 ... U P_n$ = P the model mined from P by FHM is identical to the model mined from the set of logs $P_1, P_2, ...P_n$ that are potentially distributed.

The dependency graph algorithm takes as input the dependency measure matrix. An incorrect dependency graph can only be produced in a scenario if the values contained in the final dependency matrix were incorrect.

Let us assume that the values of elements in the final dependency measures matrix are different than those required to generate the correct dependency graph. In such as case, one or more cells of the matrix will hold incorrect values. Let us further assume a matrix cell(x,y) holds an incorrect value. Since the algorithm employed to generate the local dependency measures local matrix follows the same steps as described in FHM, it is impossible to have an incorrect cell value in the locally obtained dependency measure matrix. Furthermore, the global matrix is computed by employing the secure sum protocol, ensuring that correct sum is calculated using all local dependency measures matrixes involved in the computation. Hence in an environment where a secure

communication channel exists(with no possibility of adversary attacks) and no collusion possibility, it is impossible to have a global matrix containing an incorrect cell value and consequently an incorrect dependency measures matrix. Thus, this contradicts our initial assumption that there exists a cell containing an incorrect value. Hence we conclude that the proposed distributed privacy preserving algorithm computes the correct dependency graph.

Performance: Assuming no direct connection to master site exists, in which case the nodes will form a directed cycle graph. In this case the cost to compute the global process model will be calculated by the total number of times the sites exchange messages. Analysis of the presented algorithm shows that K*C*2 number of messages will be exchanged amongst the sites to compute the global matrix. Where K is the total number of nodes. C is a constant representing the total number of times matrixes are passed around. Finally we multiply it with 2 because each site will send back an acknowledgement message back after receiving the matrix.

6 Conclusion

Distributed Process Mining remains an unexplored field. No process mining algorithms exist that can contruct process models from distributed process logs. In this thesis a novel method was presented that extends the existing Flexible Heuristic Miner algorithm to solve the problem of secure distributed process mining. The presented method allows us to construct process models from geographically dispersed process logs with minimum information sharing between the parties involved. The major contribution of this work lies in the intuition that following the principle of secure sum protocol we easily extend and convert majority of other process mining algorithms into privacy preserving distributed process mining algorithms. This is because of the fact that majority of process mining algorithms are performing similar computations(such as frequency counts etc.) at various stages. Examining the healthcare careflows to obtain meaningful knowledge is a non-trivial task given the dynamic nature of treatment processes followed in healthcare organisations [19]. The presented work has applications in healthcare sector where process logs are often dispersed amongst various healthcare organizations, which are reluctant to share them due to privacy concerns. Employing the presented method these organizations would be able cooperate and jointly mine the process logs as if they were all present in one centralized location. This will enable them to hopefully gain new insights leading to improvement of existing implemented careflows.

References

1. R.S. Mans1 , M.H. Schonenberg , M. Song , W.M.P. van der Aalst , and P.J.M. Bakker, Application of Process Mining in Healthcare A Case Study in a Dutch Hospital, Springer-Verlag Berlin Heidelberg 2008
2. Wil M.P. van der Aalst, Distributed Process Discovery and Conformance Checking, Eindhoven University of Technology, Eindhoven, The Netherlands

3. W.M.P. van der Aalst. Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer-Verlag, Berlin, 2011.
4. S. C. J.-l. A. M. Mervat Bamiah, Sarfarz Brohi, Study on significance of adopting cloud computing paradigm in healthcare sector, in Proceeding of 2012 IEEE International Conference on Cloud Computing, Technologies, Applications and Management, Dec. 2012, pp. 65-68.
5. T. S. Yunyong Guo, Mu-Hsing Kuo, Cloud computing for healthcare research information sharing, in 4th IEEE International Conference on Cloud Computing Technology and Science, Dec. 2012, pp. 889894.
6. A. E. Youssef, A framework for secure healthcare systems based on big data analytics in mobile cloud computing environments, in International Journal of Ambient Systems and Applications (IJASA), vol. 2, no. 2. June 2014.
7. J. Poelmans, G. Dedene, G. Verheyden, H. van der Mussele, S. Viaene, and E. Peters. Combining Business Process and Data Discovery Techniques for Analyzing and Improving Integrated Care Pathways. In Proceedings of ICDM10, volume 6171 of Lecture Notes in Computer Science, pages 505517. Springer-Verlag, Berlin, 2010
8. S. Gupta. Workflow and Process Mining in Healthcare. Masters thesis, Eindhoven University of Technology, Eindhoven, 2007.
9. R.S. Mans, M.H. Schonenberg, G. Leonardi, S. Panzarasa, S. Quaglini, and W.M.P. van der Aalst. Process Mining Techniques : An Application to Stroke Care. In Proceedings of MIE 2008, volume 136 of Studies in Health Technology and Informatics, pages 573578. IOS Press, 2008.
10. van der Aalst, W M P and Weijters, A J M M and Maruster, L (2003). Workflow Mining: Discovering process models from event logs, IEEE Transactions on Knowledge and Data Engineering, vol 16
11. W.M.P. van der Aalst. Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer-Verlag, Berlin, 2011.
12. Rafael Accorsi1 , Ernesto Damiani , and Wil van der Aalst, Unleashing Operational Process Mining
13. Ronny S. Mans, Wil M. P. van der Aalst, Rob J. B. Vanwersch, and Arnold J. Moleman. 2012. Process mining in healthcare: data challenges when answering frequently posed questions. In Proceedings of the 2012 international conference on Process Support and Knowledge Representation in Health Care (BPM 2012), Richard Lenz, Silvia Miksch, Mor Peleg, Manfred Reichert, David Riano, and Annette Ten Teije (Eds.). Springer-Verlag, Berlin, Heidelberg
14. R.S. Mans, M.H. Schonenberg, M.S. Song, W.M.P. van der Aalst, and P.J.M. Bakker. Application of Process Mining in Healthcare : a Case Study in a Dutch Hospital. In Proceedings of BIOSTEC 2008, volume 25 of Communications in Computer and Information Science, pages 425438. Springer-Verlag, Berlin, 2009.
15. J.Y. Zhou. Process mining : Acquiring Objective Process Information for Healthcare Process Management with the CRISP-DM Framework. Masters thesis, Eindhoven University of Technology, Eindhoven, 2009
16. L. Torres Ramos. Healthcare Process Analysis : Validation and Improvements of a Data-based Method using Process Mining and Visual Analytics. Masters thesis, Eindhoven University of Technology, Eindhoven, 2009.
17. Julia Rudnitckaia. Process Mining. Data science in action. Brno, University of Technology, Faculty of Information Technology.
18. A. Rebuge and D.R. Ferreira. Business Process Analysis in Healthcare Environments: A Methodology Based on Process Mining. Information Systems, 37(2), 2012.
19. M. Lang, T. B urkle, S. Laumann, and H.-U. Prokosch. Process Mining for Clinical Workflows: Challenges and Current Limitations. In Proceedings of MIE 2008, volume 136 of Studies in Health Technology and Informatics, pages 229234. IOS Press, 2008.
20. William Stallings. 2010. Cryptography and Network Security: Principles and Practice (5th ed.). Prentice Hall Press, Upper Saddle River, NJ, USA.
21. A.J.M.M. Weijters, W.M.P. van der Aalst, and A.K. Alves de Medeiros. Process Mining with the HeuristicsMiner-algorithm. BETA Working Paper Series, WP 166, Eindhoven University of Technology, Eindhoven, 2006.
22. A. J. M. M. Weijters and J. T. S. Ribeiro. Flexible Heuristics Miner (FHM). In CIDM, pp. 310317. IEEE, 2011.

23. Christian W. Gunther and Wil M.P. van der Aalst. Fuzzy Mining Adaptive Process Simplification Based on Multi-Perspective Metrics, Eindhoven University of Technology P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands.
24. W.M.P. van der Aalst, A.K. Alves de Medeiros, and A.J.M.M. Weijters. Genetic Process Mining,Springer Berlin Heidelberg, Department of Technology Management, Eindhoven University of Technology P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands.2005
25. Aalst WMPvd. Distributed process discovery and conformance checking. LCNS 2012; 7212:1-25.
26. PHILIP S. YU, PRIVACY-PRESERVING DATA MINING: MODELS AND ALGORITHMS, University of Illinois at Chicago, Chicago, IL 60607
27. R. Agrawal, R. Srikant, Privacy-preserving data mining, in Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 439-450, 2000.
28. ZHUOJIA XU, Analysis of Privacy Preserving Distributed Data Mining Protocols, Master thesis, School of Engineering and Science, Faculty of Health, Engineering and Science, VICTORIA UNIVERSITY. 2011
29. C. Su and K. Sakurai. Secure Computation Over Distributed Databases. IPSJ Journal, Vol. 0, No. 0, 2005.
30. Chris Clifton, Murat Kantarcioglu, Jaideep Vaidya, Xiaodong Lin, Michael Y. Zhu, Tools for Privacy Preserving Distributed Data Mining
31.]R.S. Mans1 , W.M.P. van der Aalst1 , R.J.B. Vanwersch1 , A.J. Moleman, Process Mining in Healthcare: Data Challenges when Answering Frequently Posed Questions
32. D. W. Cheung, Jiawei Han, V. T. Ng, A. W. Fu and Yongjian Fu, A fast distributed algorithm for mining association rules, Fourth International Conference on Parallel and Distributed Information Systems, 1996., , Miami Beach, FL, 1996,pp. 31-42.