An Optimal Process Model for a Real Time Process

Likewin Thomas*, Manoj Kumar M V*, Annappa B*, and Vishwanath K P†

*Department of Computer Science and Engineering
†Department of Mathematical and Computational Science
National Institute of Technology Karnataka, Surathkal,
Mangalore - 575025
India
{likewinthomas, manojmv}@nitk.ac.in
annappa@ieee.org
shastryvishwanath@gmail.com
http://www.cse.nitk.ac.in

Abstract. Recommending an optimal path of execution and a complete process model for a real time partial trace of large and complex organization is a challenge. The proposed AlfyMiner ($\alpha_y Miner$) does this recommendation in cross organization process mining technique by comparing the variants of same process encountered in different organization. $\alpha_y Miner$ proposes two novel techniques *Process Model Comparator* ($\alpha_{y Comp}$) and *Resource Behaviour Analyser* (RBA$_{Miner}$). $\alpha_{y Comp}$ identifies Next Probable Activity of the partial trace along with the complete process model of the partial trace. RBA$_{Miner}$ identifies the resources preferable for performing Next Probable Activity and analyse their behaviour based on *performance, load and queue*. $\alpha_y Miner$ does this analysis and recommend the best suitable resource for performing Next Probable Activity and process models for the real time partial trace. Experiments were conducted on process logs of CoSeLoG Project[1] and *72%* of accuracy is obtained in identifying and recommending *NPA* and the performance of resources were optimized by *59%* by decreasing their load.

Keywords: Cross Organization Process Mining, Resource Behavior, Best Resource, Polynomial Regression Model, Resource Performance, Resource Load, Resource Queue: Average Waiting Time.

1 Introduction

In the current world where the resources are being shared among different organization through the cloud computing paradigm, most of the organizations have started to shift towards Shared Business Process Management Infrastructure (SBPMI). Due to this shift in modelling paradigm, organizations have to

[1] http://dx.doi.org/10.4121/uuid:26aba40d-8b2d-435b-b5af-6d4bfbd7a270

continuously improve their process [1]. But most of the organizations are still depending on the external service providers to monitor their business process, hence the business links are to be established with those external agencies [2]. This issue was well addressed by the Information Technology by developing various work-flow tools [3] [4] [5] [6]. The challenge here is to extend the service from boundary of *single organization to cross organizations.*

Due to *data explosion* [7] getting insight and performing analysis on the data to understand their behaviour and discover an optimized process model is always been a challenge to any organization in the process mining environment. $\alpha_y Miner$ uses SBPMI, to analyse the data behaviour of an organization. This is achieved by comparing the model of same variant using RBA$_{Miner}$ in SBPMI and recommending the best suitable process model. The context of this paper is the CoSeLoG Project[2]. The data used for the experiment and analysis of proposed algorithm is obtained from the Configurable Services for Local Government (CoSeLoG) Project. This project was executed under Dutch Organization for Scientific Research (NWO) [8].

$\alpha_y Miner$ is a new analytical tool for discovering the optimal path of completion of a partial trace along with recommendation of complete process model. It proposes two novel techniques $\alpha_{y\,Comp}$ and RBA$_{Miner}$. $\alpha_{y\,Comp}$ identifies the optimal path of completion by matching the partial trace and discovering the variants in all process models logged in the repository. It identify and recommends the Next Probable Activity (NPA) of partial trace. RBA$_{Miner}$ identifies the suitable resource for performing the discovered NPA, by analysing the behaviour of all resources capable of performing NPA based on their *performance, load and waiting time.*

$\alpha_y Miner$ is analysed using the running example [2]. NPA for the partial trace and optimal process model is identified in cross organization environment using $\alpha_{y\,Comp}$ [3] and the resource preferable for performing NPA is analysed and recommended using RBA$_{Miner}$ [4]. The experiment is conducted using the real time event log of CoSeLoG Project[3] and the result of RBA$_{Miner}$ is presented in section [5].

2 Running Example

The proposed $\alpha_y Miner$ is illustrated using the running example of four variant process model containing 9 activities, shown in Figure[1b]. The corresponding sample event log describing the process execution of the process model is shown in Table[1]. Here the traces matches model perfectly which is not the cases in real life process model. The complete log file of the running example can be

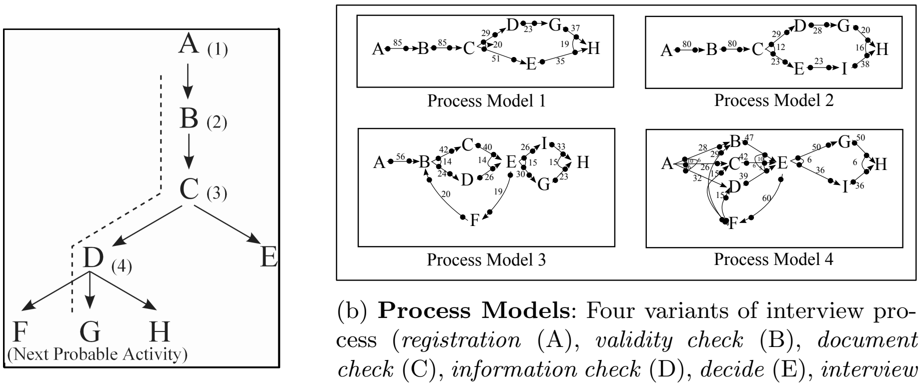
[2] http://dx.doi.org/10.4121/uuid:26aba40d-8b2d-435b-b5af-6d4bfbd7a270

[3] http://dx.doi.org/10.4121/uuid:26aba40d-8b2d-435b-b5af-6d4bfbd7a270

found at *Process Mining* @ *NITK*[4]. The experimental results are obtained using the CoSeLoG Project[5].

2.1 Proposed Problem

Consider an online process shown in Figure[1a], the dotted line shows the path of execution of the online process. Sub-scripted values at each activities are the sequence of occurrence of the activities ($A_1 \rightarrow B_2 \rightarrow C_3$). At activity C_3, decision has to be taken about which next activity to be performed, either D or E. $\alpha_y Miner$ identify the NPA and *recommends* the suitable resource for performing NPA.



(a) **Illustration of Online Process Model**

(b) **Process Models**: Four variants of interview process (*registration* (A), *validity check* (B), *document check* (C), *information check* (D), *decide* (E), *interview* (I), *group discussion* (G), *result* (H) and *re-initiates* (F))

Fig. 1: **Running Example**

3 Alfy Miner ($\boldsymbol{\alpha_y Miner}$)

$\alpha_y Miner$ is intended to identify and predict the optimal path of execution along with the complete process model, for a real time process. On identifying the currently executing activity A_i, $\alpha_y Miner$ recommends the optimal path of completion and the best suitable process model matching the *partial trace* with same variant event logs, logged in the process model repository. On identifying the matched variants, the optimal process models are identified by running *process model comparator* $\alpha_{y\,Comp}$ which matches the partial trace. Recommendation of *Next probable Activity* NPA is done by selecting NPA (A_i) in identified suitable process model. The Algorithm [1] gives the execution steps of $\alpha_y Miner$.

[4] http://http://processminingnitk.blogspot.in/2015/03/best-resource-recommendation-for.html

[5] http://dx.doi.org/10.4121/uuid:26aba40d-8b2d-435b-b5af-6d4bfbd7a270

Case ID	TRACE	Duration
10358444	$A^{12350}_{24/01/14}$ $B^{630640}_{28/01/14}$ $C^{221210}_{29/01/14}$ $D^{23640}_{02/02/14}$ $E^{7560}_{15/02/14}$ $H^{631250}_{26/02/14}$	33
12421232	$A^{23640}_{25/01/14}$ $B^{530640}_{26/01/14}$ $C^{230410}_{28/01/14}$ $D^{12350}_{09/02/14}$ $G^{7716}_{13/02/14}$ $H^{631250}_{24/02/14}$	30
12592056	$A^{12350}_{02/03/14}$ $B^{4503}_{12/03/14}$ $C^{630450}_{18/03/14}$ $G^{721560}_{26/03/14}$ $E^{7560}_{27/03/14}$ $H^{631250}_{08/04/14}$	37
12610928	$A^{23640}_{12/05/14}$ $B^{530640}_{17/05/14}$ $C^{230410}_{29/05/14}$ $E^{7560}_{05/06/14}$ $D^{23640}_{16/06/14}$ $G^{7716}_{28/06/14}$ $H^{631250}_{05/07/14}$	54
12984815	$A^{12350}_{16/08/14}$ $B^{630450}_{29/08/14}$ $C^{221210}_{09/09/14}$ $D^{12350}_{16/09/14}$ $G^{721560}_{22/09/14}$ $E^{7716}_{15/10/14}$ $H^{631250}_{27/10/14}$	72

(a) **Event Log of Process Model 1**

Case ID	TRACE	Duration
13945854	$A^{23640}_{26/01/14}$ $B^{450320}_{28/01/14}$ $C^{630450}_{31/01/14}$ $D^{23640}_{15/02/14}$ $G^{720560}_{19/02/14}$ $H^{631250}_{26/02/14}$	31
13968144	$A^{12350}_{12/02/14}$ $B^{630450}_{19/02/14}$ $C^{221210}_{22/02/14}$ $E^{12350}_{09/03/14}$ $I^{631210}_{26/03/14}$ $H^{631250}_{28/03/14}$	44
15073705	$A^{12350}_{12/04/14}$ $B^{530640}_{29/04/14}$ $C^{630450}_{02/05/14}$ $D^{12350}_{15/05/14}$ $G^{771620}_{19/05/14}$ $E^{720560}_{26/05/14}$ $H^{631250}_{08/06/14}$	57
16609162	$A^{23640}_{15/04/14}$ $B^{530640}_{19/04/14}$ $C^{230410}_{02/05/14}$ $E^{720560}_{15/05/14}$ $D^{23640}_{16/05/14}$ $G^{771620}_{18/05/14}$ $H^{631250}_{20/05/14}$	35
16789201	$A^{12350}_{19/06/14}$ $B^{630450}_{23/06/14}$ $C^{221210}_{29/06/14}$ $D^{23640}_{15/07/14}$ $G^{721560}_{27/07/14}$ $E^{771620}_{09/08/14}$ $I^{641210}_{16/08/14}$ $H^{631250}_{23/08/14}$	65

(b) **Event Log of Process Model 2**

Case ID	TRACE	Duration
16796450	$A^{12350}_{02/05/14}$ $B^{450320}_{23/05/14}$ $C^{630450}_{15/06/14}$ $E^{720560}_{19/06/14}$ $I^{651210}_{09/07/14}$ $H^{631250}_{27/07/14}$	86
17031584	$A^{23640}_{26/07/14}$ $B^{450320}_{15/08/14}$ $C^{221210}_{29/08/14}$ $E^{720560}_{12/09/14}$ $F^{720560}_{28/09-14}$ $B^{630450}_{13/10/14}$ $C^{221210}_{18/10/14}$ $E^{720560}_{22/10/14}$ $I^{651210}_{29/10/14}$ $H^{631250}_{30/10/14}$	96
17939005	$A^{12350}_{05/10/14}$ $B^{630450}_{13/10/14}$ $C^{630450}_{22/10/14}$ $E^{720560}_{29/10/14}$ $F^{720560}_{13/11/14}$ $B^{450320}_{19/11/14}$ $D^{23640}_{02/12/14}$ $E^{720560}_{06/12/14}$ $G^{720560}_{10/12/14}$ $H^{631250}_{24/12/14}$	80
19472044	$A^{23640}_{15/12/14}$ $B^{530640}_{19/12/14}$ $C^{630450}_{28/12/14}$ $E^{720560}_{03/01/15}$ $F^{720560}_{05/01/15}$ $B^{630450}_{16/01/15}$ $C^{230410}_{18/01/15}$ $E^{720560}_{22/01/15}$ $G^{721560}_{23/01/15}$ $I^{631210}_{28/01/15}$ $H^{631250}_{29/01/15}$	45
25845687	$A^{23640}_{12/11/14}$ $B^{530640}_{14/12/14}$ $C^{630450}_{19/12/14}$ $E^{720560}_{22/12/14}$ $G^{721560}_{27/12/14}$ $H^{631250}_{30/12/14}$	48

(c) **Event Log of Process Model 3**

Case ID	TRACE	Duration
19830478	$A^{12350}_{02/05/14}$ $B^{12350}_{02/05/14}$ $E^{12350}_{02/05/14}$ $G^{12350}_{02/05/14}$ $H^{12350}_{02/05/14}$	53
19834032	$A^{12350}_{02/05/14}$ $B^{12350}_{02/05/14}$ $E^{12350}_{02/05/14}$ $F^{12350}_{02/05/14}$ $C^{12350}_{02/05/14}$ $E^{12350}_{02/05/14}$ $G^{12350}_{02/05/14}$ $H^{12350}_{02/05/14}$	52
19836934	$A^{12350}_{02/05/14}$ $B^{12350}_{02/05/14}$ $C^{12350}_{02/05/14}$ $E^{12350}_{02/05/14}$ $F^{12350}_{02/05/14}$ $B^{12350}_{02/05/14}$ $E^{12350}_{02/05/14}$ $G^{12350}_{02/05/14}$ $H^{12350}_{02/05/14}$	59
19838656	$A^{12350}_{02/05/14}$ $D^{12350}_{02/05/14}$ $B^{12350}_{02/05/14}$ $E^{12350}_{02/05/14}$ $F^{12350}_{02/05/14}$ $B^{12350}_{02/05/14}$ $E^{12350}_{02/05/14}$ $G^{12350}_{02/05/14}$ $H^{12350}_{02/05/14}$	37
19844185	$A^{12350}_{02/05/14}$ $D^{12350}_{02/05/14}$ $C^{12350}_{02/05/14}$ $E^{12350}_{02/05/14}$ $F^{12350}_{02/05/14}$ $B^{12350}_{02/05/14}$ $E^{12350}_{02/05/14}$ $G^{12350}_{02/05/14}$ $H^{12350}_{02/05/14}$	29

(d) **Event Log of Process Model 4**

Table 1: **Event logs of four different process models of interview process** shown in figure[1b]. Each log table shows Case ID_1, $Trace_2$ and the total $duration_3$. Each cell in trace, shows the activity of the trace, *Resource* (Superscripted) and the time of occurrence of that activity (sub-scripted).

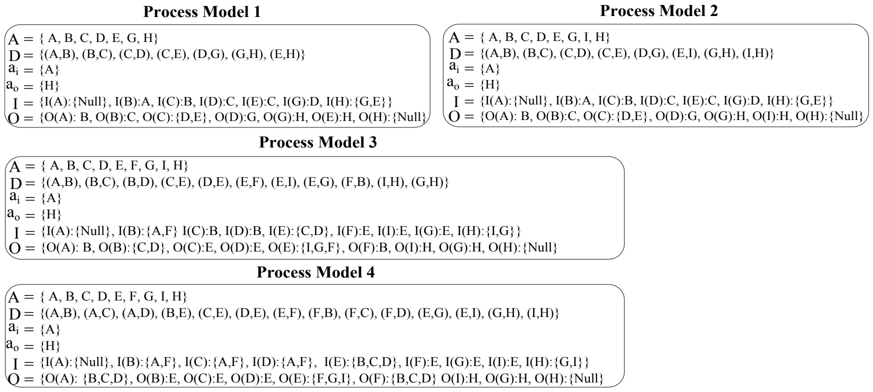
Algorithm 1: $\alpha_y Miner$

```
Input: Partial Real Time Trace
Output: NPA & Process Model
Develop Process model repository;
repeat
    $Match_{Var}$ ← Call Match Variant($A_i$);
    $\alpha_y Comp$ ← $\alpha_y$ Comp($Match_{Var}$) ;
    Set(NPA) ← $InOut_{Binding}$(C-Net )
until for each currently executing activity $A_i$
```

3.1 Process Model: Casual Net

$\alpha_y Miner$ uses Casual Net: C-Net notation to represent the process model. C-Net is a six-tuple: {A,D,a_i,a_o,I,O} representation of process model with *A*:{set of activities}, *D*:{Set of Dependencies}, a_i:{Set of Start activities}, a_o: {Set of Output activities} , *I*: {Set of Input Binding} , *O*: {Set of Output Binding}.

C-Net for all the four process model of the running example is shown in Figure 2. The repository of process model is maintained for analysing process behaviour.



Process Model 1

A = { A, B, C, D, E, G, H}
D = {(A,B), (B,C), (C,D), (C,E), (D,G), (G,H), (E,H)}
a_i = {A}
a_o = {H}
I = {I(A):{Null}, I(B):A, I(C):B, I(D):C, I(E):C, I(G):D, I(H):{G,E}}
O = {O(A): B, O(B):C, O(C):{D,E}, O(D):G, O(G):H, O(E):H, O(H):{Null}

Process Model 2

A = { A, B, C, D, E, G, I, H}
D = {(A,B), (B,C), (C,D), (C,E), (D,G), (E,I), (G,H), (I,H)}
a_i = {A}
a_o = {H}
I = {I(A):{Null}, I(B):A, I(C):B, I(D):C, I(E):C, I(G):D, I(H):{G,E}}
O = {O(A): B, O(B):C, O(C):{D,E}, O(D):G, O(G):H, O(I):H, O(H):{Null}

Process Model 3

A = { A, B, C, D, E, F, G, I, H}
D = {(A,B), (B,C), (B,D), (C,E), (D,E), (E,F), (E,I), (E,G), (F,B), (I,H), (G,H)}
a_i = {A}
a_o = {H}
I = {I(A):{Null}, I(B):{A,F} I(C):B, I(D):B, I(E):{C,D}, I(F):E, I(I):E, I(G):E, I(H):{I,G}}
O = {O(A): B, O(B):{C,D}, O(C):E, O(D):E, O(E):{I,G,F}, O(F):B, O(I):H, O(G):H, O(H):{Null}

Process Model 4

A = { A, B, C, D, E, F, G, I, H}
D = {(A,B), (A,C), (A,D), (B,E), (C,E), (D,E), (E,F), (F,B), (F,C), (F,D), (E,G), (E,I), (G,H), (I,H)}
a_i = {A}
a_o = {H}
I = {I(A):{Null}, I(B):{A,F}, I(C):{A,F}, I(D):{A,F}, I(E):{B,C,D}, I(F):E, I(G):E, I(I):E, I(H):{G,I}}
O = {O(A): {B,C,D}, O(B):E, O(C):E, O(D):E, O(E):{F,G,I}, O(F):{B,C,D} O(I):H, O(G):H, O(H):{Null}

Fig. 2: **C-Net Representation of process Model in Figure** 1b

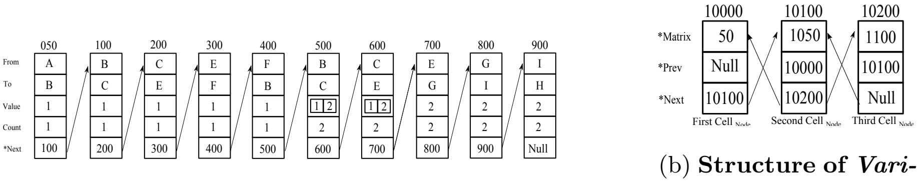
3.2 Matching variants with Path Detector

When an online process is getting executed, identifying to which variant the currently executing trace belongs is a challenge for $\alpha_y Miner$. Algorithm Variant $_{Match}$[2] identify the path of execution along with the set of possible NPA. Variant$_{Match}$ uses the concept of linked list with 2 nodes: *Cell*$_{Node}$ and *Variant*$_{Node}$ which are represented as *class*. *Cell*$_{Node}$ $= \{from_1 \leftarrow \bigcup\{\bullet a\}$, to$_2 \leftarrow a$, $value_3 \leftarrow \{|\bullet a \rightarrow a|\sigma\}$, $count_4 = |\bullet a \rightarrow a| \in \zeta$. *Variant*$_{Node}$ $\{$**matrix* (address of *Cell*$_{Node}$), **prev*$_2$ **next*$_3$ (address of next and previous *Cell*$_{Node}$)$\}$. The *Cell*$_{Node}$ Figure[3a] stores the information of trace A→B→C→E→F→B→D→E→G→H of process model 2. The *value*$_3$ field remains 1 till the sequence in trace appears first time. On identifying the *loop*, value in *value*$_3$ filed is updated to 2 as shown at *Cell*$_{Node}$ with memory 500 in Figure[3a]. *Value*$_3$ field is an array and stores the value 1,2 to indicate the sequence B→C is appearing second time in the trace.*Count*$_3$ is a counter of the sequence appearance in the trace. *Variant*$_{Node}$ Figure[3b] stores the information of all the variants. This is used while comparing the online sequence with the variants. *If* a variant matches the sequence, *then* that variant is retained *else* it is deleted from the linked list.

3.3 Process Model Comparator ($\alpha_{y\,Comp}$)

$\alpha_{y\,Comp}$ compares the C-Net of all the variants in cross organization environment based on following comparison metrics.

1. *Process Model Metric*: Compare total number of activities, resources, traces and variants
2. *Relation Metric*: Compare total number of parallel, serial activities and loops.



(a) **Structure of *Cell***$_{Node}$ for sequence A→B→C→E→F→B→D→E→G→H of process model 2

(b) **Structure of *Variant***$_{Node}$ for the set of *Cell*$_{Node}$ of process model 2

Fig. 3: **Structure of Cell**$_{Node}$ **and Variant**$_{Node}$

Algorithm 2: Matching the Variants: $Variant_{Match}()$

```
   Input: Online process
   Output: Matching matrix
1  Match Variant() struct variant$_{Node}$⋆gvn, ⋆tempvn; (gvn : address of linked list say
   globle Variant Node), Let ⋆gvn gives address of the double linked list, Initialize all counter
   in cell$_{Node}$ → 0;
2  repeat
3      ⋆tempvn ← &gvn Get the address of the double linked list;
4      repeat
5          ⋆tempcn ← &matrix Get the address of the matrix;
6          tempcn→from = sequence[i] ∧ tempcn→to = sequence[i+1];
7          if not found then Delete current variant$_{Node}$ from double linked list and go to 5
8          else Increment the member variable count;
9          if count == val[count] (Current and previous check are passed) then Go to
           next→variant$_{Node}$ in the double linked list and go to step 5
10         else Delete the current→variant$_{Node}$ from the double linked list and go to 5
11     until ⋆next in double linked list is null
12 until for each activity in online process
13 Remaining variant$_{node}$ present in tempvn are all matched variant table for the given
   sequence.
```

3. *Complexity Metric*: Compare total number of split and join.
4. *Service Time Metric*: Compare the queue time for each activity.
5. *Fitness Metric*: Running fitness test along with the time of completion and valid no of sequence in each event log.

Process Model Metric The process model comparison is done based on No of {Activities, Resources, Traces & Varinats } and is shown in Table 2a.

Relation Metric $\alpha_{y_{Comp}}$ analysed that if a model has more parallel relation it performs well when compared to serial relation, at the same time if the loop is increased the consumption of execution time also increases. Parallel relation is identified by Equation 4 in Definition 1. Loops are identified by Equation 5.

Definition 1. ***Log based ordering relation***
Let $\mathscr{A}$ = [a, b, c, d, e] be the set of activities and let L be the simple event log

i.e., $L \in \mathscr{A}^{\,*}$ *and Let A be* a_i^{th} *activity and B be* a_{i+1}^{th} *then,*

$$\dot{\mathit{DirectlyFollow}}_{(a>_L b)} \leftarrow \{\mathit{iff}\ \ \exists\ \ \mathit{trace}\ \ \sigma = \langle t_1, t_2, ..., t_n\rangle \wedge$$

$$i \in [1, 2, ..., n-1] \mid \sigma \in L,$$

$$\wedge\ \ t_i = a, \wedge\ t_i + 1 = b\} \tag{1}$$

$$\dot{\mathit{Casuality}}_{(a \longrightarrow_L b)} \leftarrow \{\mathit{iff}\ \ a >_L b \wedge b \not>_L a\} \tag{2}$$

$$\dot{\mathit{Unrelated}}_{(a \#_L b)} \leftarrow \{\mathit{iff}\ \ a \not>_L b \wedge b \not>_L a\} \tag{3}$$

$$\dot{\mathit{Parallel}}_{(a \|_L b)} \leftarrow \{\mathit{iff}\ \ a >_L b \wedge b >_L a\} \tag{4}$$

$$\dot{\mathit{Loop}}_{(a >_L b >_L a)} \leftarrow \{\mathit{iff}\ (a_i\ \ ==\ \ a_{i+2})\ \ \rightarrow\ \ a_i >_L a_{i+1} >_L a_{i+2}\} \tag{5}$$

The Table 2b gives the relation metric of all the four models in running example.

Complexity Metric Complexity metric identifies the joins and splits in the process model. Joins and split are identified using the result of output and input binding. Consider the Figure[1b] where for process model 1: O(A)={B}=85 times, similarly the *split* {CDE} = 20, its means 20 times activity C is 20 times followed by both D and E, *join* {GEH} is joined 16 times. Using this information complexity metric shown in Table[2c] is developed.

Service Time Metric This metric gives the total service time comparison for an activity in each model. This comparison helps in identifying the model serving an activity with less service time. The service time is calculated by $\sum_{i=1}^{each\ cases} duration(A_i)$, where A$_i \subseteq \mathscr{A}(set\ of\ activities)$. The sample output in seconds is shown in Table 2d.

Fitness Metric This gives the numbers of traces that can be successfully run on the model. This is helpful in deciding how efficient the model is, in running the trace. $\alpha_{y_{Comp}}$ identifies the model which runs maximum number of traces with minimum time. Consider the Table 2e.

3.4 Binding Relation

On identifying variants following the partial trace, the NPA of currently executing activity A$_i$ is identified using binding relation which bind the incoming and outgoing activity of A$_i$. Algorithm 3 eplain the concept of binding relation, where for each trace in a case, *if* an activity A is followed by B, *then* A.outbond ← B $\wedge$ B.inbound ← A, i.e., A has *out-bounding relationship* with B and similarly B as *in-bounding relationship* with A

	No of Activities	No of Resources	No of Traces	No of Variants
PM 1	8	16	90	10
PM 2	8	14	80	13
PM 3	9	14	56	19
PM 4	9	14	86	51

(a) **Process Model Metric**

	No of Dependency	No of Parallel	No of Loops	No of Serial
PM1	7	2	0	5
PM2	8	4	0	4
PM3	11	2	2	7
PM4	14	3	3	

(b) **Relation Metric**

	Joins	Splits
PM1	19	20
PM2	16	12
PM3	29	29
PM4	33	28

(c) **Complexity Metric**

	A	B	C	D
PM1	3678956	45896374	56987845	1236589
PM2	2598964	56978746	78594785	4589647
PM3	4577896	36987567	23698124	5698347
PM4	1236978	23678945	22456378	4548768

(d) **Service Time Metric**

	PM1	T(PM1)	PM2	T(PM2)	PM3	T(PM3)	PM4	T(PM4)
Event Log1	1	56897845	0.9	78456975	0.75	45789647	0.65	56587874
Event Log2	0.8	45878123	1	45678412	0.9	78956478	0.95	78945698
Event Log3	0.6	45236984	0.75	56898774	1	69875457	1	65327841
Event Log4	0.45	32789564	0.6	68974564	0.75	39845641	1	

(e) **Fitness Metric**

Table 2: **Process Model Comparator** ($\alpha_{y\,Comp}$)

Algorithm 3: To calculate Input & Output Binding

```
InOut_Binding() Input: $A_i$, RTrace
Output: $A_i.Input_{Binding}, A_i.Output_{Binding}$
repeat
    if ($|a >_L b|$) then
        a.Outbound ← $b \wedge b.Inbound \leftarrow$ a
    $|a >_L b| = \sum_{\sigma \in L} L(\sigma) \times |\{1 \leq i < |\sigma| \mid \sigma(i) = a \wedge \sigma(i+1) = b\}|$ [see [7]]
until for each sequence in trace $\sigma$ in event log L
```

4 Resource Behaviour Analyser (RBA$_{Miner}$)

$\alpha_y Miner$ on discovering suitable process model with NPA identifies the resources preferable for performing NPA. Set of resource preferable for performing NPA is identified using Activity/Resource$_{rep}$[3]. RBA$_{Miner}$ analyse the behaviour and recommend the suitable resource for performing NPA. Behaviour of the resources is analysed based on 3 parameter: *Performance, Load and Queue* using polynomial regression model for load and performance [4.2] and Average Servicing Time at resource using queue model [4.3]. Algorithm 4 explains the concept of resource behaviour analysis.

4.1 Activity/Resource$_{rep}$

$\alpha_y Miner$ identifies the list of resources performing an activity in entire process log along with the time consumed by them for performing that activity. The Table 3 gives representational view of list of resources performing an activity in process model 1 along with the time consumed.

Algorithm 4: RBA$_{Miner}$

```
$RBA(NPA)()$
Input: $NPA \& BestRes_{Activity}$
Output: $Recommendation of Res_{(NPA)}$
repeat
    $Load(Res_{(NPA)}) \longleftarrow Poly.Load(Load(Res_{(NPA)}))$; [see algo5]
    $Perf(Res_{(NPA)}) \longleftarrow Poly.Perf(Res_{(NPA)})$; [see algo5]
    $AvgWaiting_Time(Res_{(NPA)}) \longleftarrow Queue(Res_{(NPA)})$; [see algo 6]
until (for each resource of $NPA$ in $BestRes_{Activity}$ Table)
Recommend the optimal load, performance and waiting time resource
```

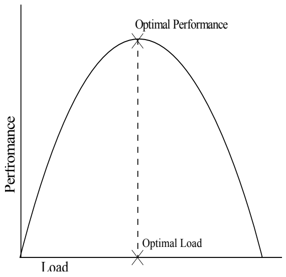
Activity	Res_{12350}	Res_{23640}	Res_{630450}	Res_{530640}	Res_{450320}	Res_{221210}	Res_{230410}	Res_{501}	Res_{771620}	Res_{502}	Res_{771620}	Res_{721560}
A	36.657	45.380	DNP	DNP	DNP	DNP	DNP	DNP	DNP	DNP	DNP	DNP
B	DNP	DNP	18.473	22.667	9.25	DNP	DNP	DNP	DNP	DNP	DNP	DNP
C	DNP	7	24.684	DNP	DNP	5.4667	22.294	DNP	DNP	DNP	DNP	DNP
D	DNP	25.53	DNP	DNP	DNP	DNP	DNP	72	11.5	DNP	DNP	DNP
E	DNP	25.531	DNP	DNP	DNP	DNP	DNP	62.5	DNP	91	11.5	DNP
G	DNP	DNP	DNP	DNP	DNP	DNP	DNP	DNP	7	DNP	DNP	13.944

Table 3: **Activity/Resource$_{rep}$ of process model 1 of running example** [DNP: *Did Not Play*]

4.2 Resource load & performance analyser

The *Yerkes-Dodson Law of Arousal*, also known as *Arousal Theory*, states that by increasing arousal, the workers performance can be improved. However, if the level of arousal increases too much, performance decreases Figure[4a] [9]. The RBA$_{Miner}$ identifies the *level of arousal*: *Optimal Load* i.e., the *maximum load* the resource can handle efficiently, along with its *performance* using *polynomial regression model.* Performance is a ratio of *Total time taken* by *Load.* The performance was analysed by increasing the load and observing the time taken. It was observed that, as the load was increased, the consumption of the time was decreasing. But at some point there was a drift and the time consumption started increasing. That drifted point is known as Arousal (optimal load and performance of the resources). The Algorithm[5] identifies the load ℓ and performance [$Total\ time \div \ell$] for /resource/unit time.

The RBA$_{Miner}$ first *filters* the *unperformed load*[1] (an activity with 0 ms) and *residual load*[2] (an activities with exceptional duration). Then the *actual load* (ℓ) and *average time of Service* (α) of each worker each month is identified. Polynomial regression model[5] is applied on this cleaned data. Since the RBA$_{Miner}$ is intended in identifying the second degree regression model, the regression model initialize a 3×3 matrix (A) and 3×1 matrix (B) as shown in figure [4b& 4c]. Then the transpose of matrix A is multiplied with matrix B. The result obtained is the coefficient of polynomial equation. On applying the load on an equation the polynomial curve (power curve) is obtained as shown in figure. On analysing the polynomial curve and applying the Yerkes-Dodson Law the *optimal load* and *optimal performance* of a resource is identified for each month.



$$A = \begin{bmatrix} n & \sum_{1}^{n} \ell & \sum_{1}^{n} \ell^2 \\ \sum_{1}^{n} \ell & \sum_{1}^{n} \ell^2 & \sum_{1}^{n} \ell^3 \\ \sum_{1}^{n} \ell^2 & \sum_{1}^{n} \ell^3 & \sum_{1}^{n} \ell^4 \end{bmatrix}$$

$$B = \begin{bmatrix} \sum_{1}^{n} \alpha \\ \sum_{1}^{n} (\alpha \times \ell) \\ \sum_{1}^{n} (\alpha \times \ell^2) \end{bmatrix}$$

(a) **Yerkes Dodson Law** (b) **Matrix Table A** (c) **Matrix Table B**

Fig. 4: **Structure of Power Curve** for identifying the Optimal Load and Performance and the Structure Initial load & performance matrix for running Polynomial Regression Model

Algorithm 5: Resource *Second Order* Polynomial Regression Model

```
Input: $\ell$ (Total Load) on each resource each month and $\alpha$ (Log(Average Service Time)) for
        running the load $\ell$ per month
Output: Optimal Load $\mathscr{L}$ & Optimal Performance $\mathscr{P}$
Let A[3,3] & B[1,3] be 2 initial Matrix as shown in figure[4b & 4c]; k=0;
repeat
    $A^{Inverse} \longleftarrow Transpose(A,3)$; Transpose: Function transposing the matrix;
    Result $\longleftarrow$ multiplyMatrices($A^{Inverse}$, B); multiplyMatrices: Function for
    multiplying matrix;
    repeat
        $\beta[i] \longleftarrow Result[i][j]$; where $\beta$= Coefficient of Polynomial Equation
    until ((i=0 to 3) $\wedge$ j=0)
    Polynomial Equation : $\beta_0 + \beta_1\ell + \beta_2\ell^2$
until (for each resource each unit time)
```

4.3 Activity Servicing Time Model

Along with identification of load and performance of the resource preferable for performing NPA, RBA$_{Miner}$ also finds the *Activity Servicing Time* (i.e., the average waiting time for an activity to be served by a resource), before that resource is recommended. Since the interest is in finding the queue at each resource, RBA$_{Miner}$ uses Single-Server Models *(M/M/1):(GD/∞/∞)* and *(M/M/1):(GD/N/∞)*. Here the model (M/M/1):(GD/∞/∞) describe *(Arrival*^{1}*/ Departure*^{2}*/ Server*^{3}*):(Queue discipline*^{4}*/ Max number in Queue*^{5}*/ Source of Calling*^{6}*)*.

Arrival^1 (λ) is the rate at which the activities are arrived at each resources and *Departure*^{2} (μ) is the rate at which the arrived activities are served. Since RBA$_{Miner}$ is intended in identifying the *average waiting time* at each resource, the single server model is applied. When data was analyzed for First Come First Serve *FCFS*, Last Come First Serve *LCFS* and Service in Random Order *SIRO*, it was understood that arrival of the activity was following General Discipline *GD* as its *Queue Discipline*^{4}. As the number in queue and source of calling is not defined RBA$_{Miner}$ marks them as *infinity.* The average waiting time in the system W_s is identified using Equations [6- 9]. The Algorithm Activity Servicing Time [6] starts with identifying the arrival rate λ and the servicing rate μ at each resources.

The λ_n & μ_n in generalized model is shown in Equation[6]. The traffic ρ: number of activities arriving and getting served per unit time is shown in Equation[7]. Hence the Average waiting time in system L_s is given in Equation[9].

$$\left.\begin{aligned}\lambda_n &= \lambda \\ \mu_n &= \mu\end{aligned}\right\} \quad \textit{Where } n = \textit{0,1,2....} \tag{6}$$

$$\rho = \frac{\lambda}{\mu} \tag{7}$$

$$W_s = \frac{L_s}{\lambda} \tag{8}$$

$$L_s = \frac{\rho}{1-\rho} \tag{9}$$

Algorithm 6: To Discover the Activity Servicing Time

```
Input: $\textit{Set of resources}:\Re$, $\textit{Trace}:\Im$, $\textit{Duration of service}:\partial$
Output: Arrival $\lambda$, Service $\mu$, Traffic $\rho$, $L_s, W_s$
Let $\textit{Arrival } \lambda \in$ Load $\ell$ discovered on /Resource/month; Service $\mu \in$ service rate of $\lambda$; $\mathit{\Pi}$ be No of Days in month
if $(\textit{if } ((\mathit{\Pi} - \Im.\textit{Date}) \times 24hrs \times 60Sec) \geq \Im.\partial$ then  Event is executed in same month;
    $\mu(\Re_{Filtered}\textit{-Year\_Month}) \longleftarrow \mu(\Re_{Filtered}\textit{-Year\_Month}) + 1$;
else   $\perp = \lceil \frac{((\Im.\partial) - ((\mathit{\Pi} - \Im.Dt) \times 24hrs \times 60Sec))}{\mathit{\Pi} \times 24 \times 60} \rceil$
    $\mu(\Re_{Filtered}\textit{-Year\_Month} + \perp) \longleftarrow \mu(\Re_{Filtered}\textit{-Year\_Month} + \perp) + 1$;
$\textit{Average Servicing Time in system} \leftarrow$ Equation [6 to 9]
```

5 Experimental Analysis and Result

The $\alpha_y Miner$ algorithm is evaluated by running it on CoSeLoG Project[6]. The experiments *Exp*$_{NPA}$, *Exp*$_{AST}$ and *Exp*$_{L\&P}$ was performed on the CoSeLoG Municipality 2, which contains 645 cases and 376 activities. Experiments were conducted and analysed on set of every 100 cases. $\alpha_y Miner$ makes 4 *assumption*: Any activity whose duration is recorded as 0 millisecond is considered as never been executed, since the nanosecond time is not recorded, vocabulary of an activity is not taken into account [1], don't deal with Live or Dead locks and assume that all process have same starting activity.

5.1 Design of Experiment

The $\alpha_y Miner$ experimental set up is shown in Figure [5]. Where the log is first cleaned and initialized using initializer from which the NPA is identified. Optimal resource for performing NPA is identified and their behaviour is analysed. Finally $\alpha_y Miner$ recommends the best *process and resource model.*

[6] http://dx.doi.org/10.4121/uuid:26aba40d-8b2d-435b-b5af-6d4bfbd7a270

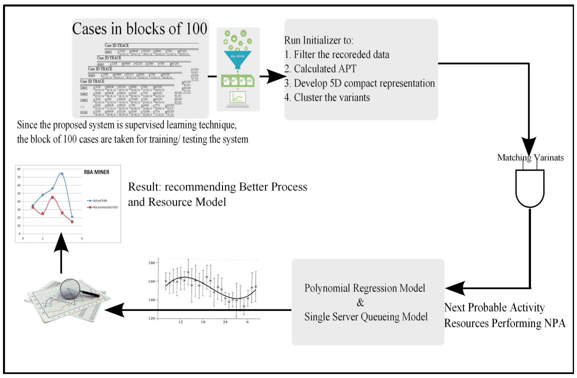

Cases in blocks of 100
Case ID TRACE
RBA MINER
Run Initializer to:
1. Filter the recoreded data
2. Calculated APT
3. Develop 5D compact representation
4. Cluster the variants
Since the proposed system is supervised learning technique,
the block of 100 cases are taken for training/ testing the system
Matching Varinats
RBA MINER
Actual Path
Recommended Path
Result: recommending Better Process
and Resource Model
Polynomial Regression Model
&
Single Server Queueing Model
Next Probable Activity
Resources Performing NPA

Fig. 5: **Illustration of Online Process Model**

5.2 Recommendation of Next Probable Activity (NPA): $\boldsymbol{Exp_{NPA}}$

Experiment was simulated in the form of supervised learning, where the test Exp_{NPA} was conducted for every 100 cases and starting from 2^{nd} activity of the sequence. Exp_{NPA} was analysed by comparing it with the actual path of execution. The result of this comparison is shown un Figure [6] and on analysis it is studied that the percentage of error rate (*marked by green line*) in recommendation is lesser in later positions of execution when compared to earlier positions. The Exp_{NPA} achieved *72.8568%* of efficiency. On analysing the graph, it is understood that the behaviour of recommended path is always below the actual path of execution. Inclination shows the huge difference of behaviour between the actual and recommended path. For the cases 400 to 500, it is observed that the graph don't have red line, as the path of execution is critical and was observed to take optimal time for completion. Hence this proves that $\alpha_y Miner$, don't recommend if the path of execution is observed to be optimal.

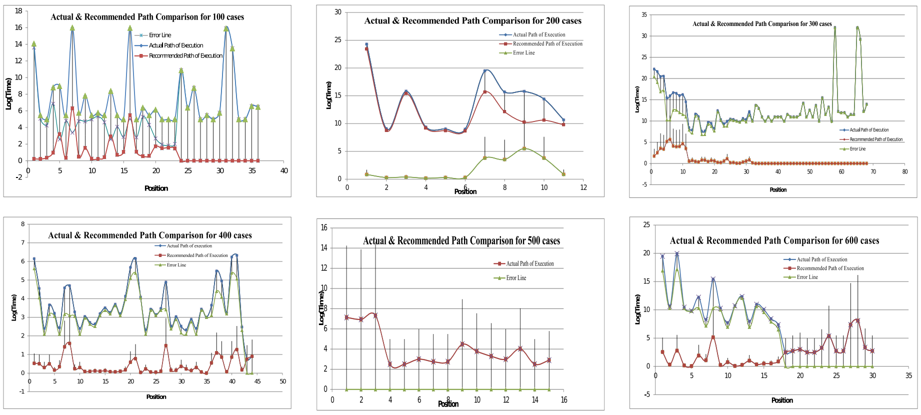


Fig. 6: **Result of Exp**$_{NPA}$

5.3 Recommendation of Resource capable for performing NPA: Exp_{AST}

The Exp_{AST} for each resource performing NPA. Waiting time of recommended resource was compared with the actual resource and it was studied that their performance was improved by *59.7303%*. The Figure [7] show the result of Exp_{AST}. The Exp$_{AST}$, discovered the better path of execution based on resource average service time and it is also understood $\alpha_y Miner$, don't recommend if the resources to whom the task is assigned is efficient in performing.

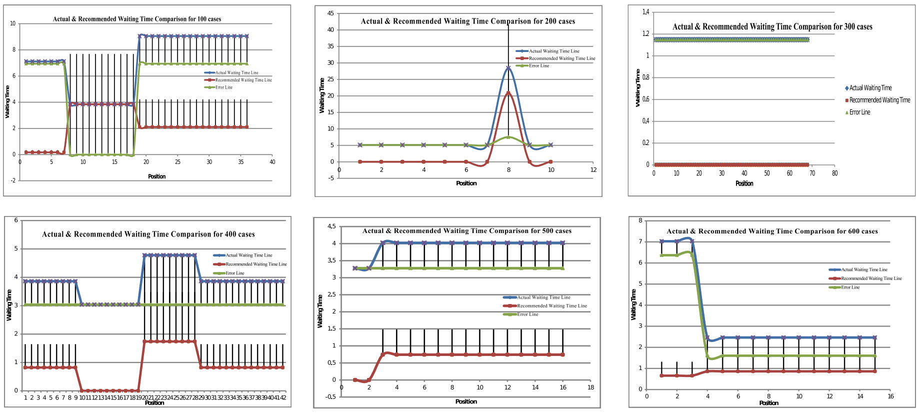


Fig. 7: **Result of Exp$_{AST}$**

	100	200	300	400	500	600	Overall
560530	0.0178571	0.005128	0.005525	0.006329	0.005495	0.009009	0.000787
560598	0.1666667	0.083333	0.166667	0.333333	0.111111	0.090909	0.016949
560521	0.0714286	0.090909	0.083333	0.012987	0.052632	0.008621	0.004587
560532	0.0076336	0.005102	0.009009	0.007194	0.003279	0.005051	0.000517
4634935	0.1428571	0.083333	0.142857	0.043478	0.009709	0.016129	0.006329
560458	0.0069444	0.007519	-0.00115	0.00304	0.00625	0.006369	0.00036
560429	0	1	1	1	1	1	1
560528	0	1	0.5	1	0.5	1	0.166667
560519	0.0153846	0.009009	0.013699	0.01	0.007246	0.001605	0.001754

Table 4: **Result of Average Waiting time for CoSeLoG project**

5.4 Polynomial regression model: $Exp_{L\&P}$

The result of $Exp_{L\&P}$ is shown in Table [5] and the Figure [8] shows the polynomial curve. Using the law of *Arousal*, the optimal load and performance at each resource can be identified. This result is used in making appropriate decision about resource behaviour and load assignments. Using the outcome of experiment proper recommendations can be made, whether to assign the task to that resource ot not.

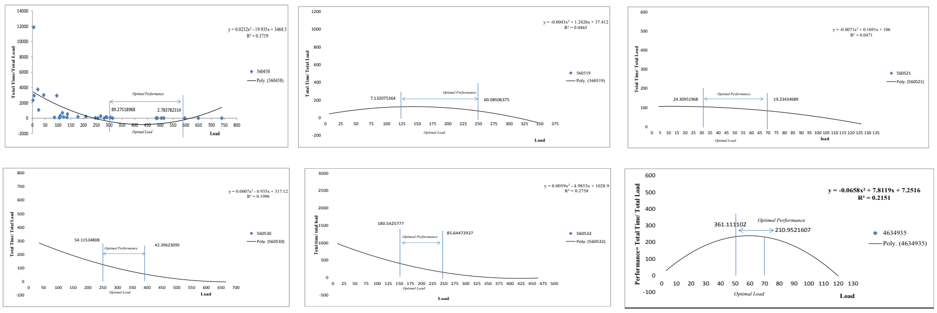


Fig. 8: **Result of** $\boldsymbol{Experiment}_{Load\&Performance}$

Resources	**No of load**	**total time**	**R^2**	**Load Range**	**Performance Range**
4634935	744	124351.8	0.2151	50-70	210.9521607 - 361.111102
560458	7838	1161852	0.3719	300-600	2.783782314 - 89.27519
560519	4809	390477.9	0.0465	125-250	7.132075 - 60.08506
560521	1475	92446.43	0.0471	30-70	19.23435 - 24.30952
560530	11140	905091.6	0.1996	250-400	42.39623 - 54.11535
560532	7817	1221671	0.2758	150-250	180.5426 - 85.64473

Table 5: **Result of Polynomial Regression for CoSeLoG project**

6 Conclusion

$\alpha_y Miner$ provided a solution for *recommending* an optimal *path of execution: NPA* along with the *complete process model* and *resource preferable for performing NPA*. $\alpha_y Miner$ is a analytical tool which gave solution for real time business process execution, by analysing the process and resource behaviour. The Experimental result shows 72% of optimization in *process execution* and 59% improvement in the behaviour of resource based on their *Average waiting time, load and performance*. $\alpha_y Miner$ was successful in *recommending appropriate process and resource model* for the real time process.

References

1. Joos CAM Buijs, Boudewijn F van Dongen, and Wil MP van der Aalst. Towards cross-organizational process mining in collections of process models and their executions. In *Business Process Management Workshops*, pages 2–13. Springer, 2012.

2. Justus Klingemann, Jurgen Wasch, and Karl Aberer. Deriving service models in cross-organizational workflows. In *Research Issues on Data Engineering: Information Technology for Virtual Enterprises, 1999. RIDE-VE'99. Proceedings., Ninth International Workshop on*, pages 100–107. IEEE, 1999.
3. Diimitrios Georgakopoulos, Mark Hornick, and Amit Sheth. An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and parallel Databases*, 3(2):119–153, 1995.
4. Gustavo Alonso, Divyakant Agrawal, Amr El Abbadi, and Carl Mohan. Functionality and limitations of current workflow management systems. *IEEE Expert*, 12(5):105–111, 1997.
5. Asuman Dogac. *Workflow management systems and interoperability*. Number 164. Springer Science & Business Media, 1998.
6. Andrzej Cichocki. *Workflow and process automation: concepts and technology*. Springer Science & Business Media, 1998.
7. Wil Van Der Aalst. *Process mining: discovery, conformance and enhancement of business processes*. Springer Science & Business Media, 2011.
8. J.C.A.M.; Buijs. Environmental permit application process (wabo), coselog project, 2014.
9. Joyce Nakatumba and Wil MP van der Aalst. Analyzing resource behavior using process mining. In *Business Process Management Workshops*, pages 69–80. Springer, 2010.