Process Mining Assignment 2 & 3: Implementing the Alpha Algorithm

Course Instructor
Sir Irfan

Submitted By

Khadija Khalid (22i-2557)

Members:

Khadija Khalid (22i-2557)

Hassaan (22i-2434)

Sana Urooj (22i-2456)

Iman (22i-2572)

Yahya Rashid (22i-1413)

Date:
24-11-2024

Fall 2024

Department of Computer Science

FAST – National University of Computer & Emerging Sciences

Table of Contents

Introduction **3**
Task 1 (ASSIGNMENT 02): EVENT LOG GENERATION PROGRAM **3**
Steps 3
Task 2 (ASSIGNMENT 03): IMPLEMENTING THE ALPHA ALGORITHM **3**
1. Step1: Display Final Log (L) 4
2. Step2: Extract and Sort Unique Events (TL), Initial Events (TI), and Final Events (TO) 4
● Unique Event (TL): A list of all unique events from the event log 4
● Initial Event (TI): The first event in unique event list 4
● Final Event (TO): The last event in unique event list 4
3. Step 3: Determining Relationships (Causal, Parallel, Choice) 5
● Causal Relationships: 5
● Parallel Relationships: 5
● Choice Relationships: 5
4. Step 4: Footprint Matrix 7
5. Step 5: Generate Pair Sets of Causal Relationships (XL) 8
6. Step 6: Displaying Maximal Relationship Pairs (YL) 9
7. Step 7: Place Set (PL) for Maximal Causal Pairs 10
8. Step 8: Flow Relation (FL) for Maximal Causal Pairs 11
Task 3 (ASSIGNMENT 02): VISUALIZATION OF THE PROCESS MODEL **12**
Petri Net: 13
Task 4 (ASSIGNMENT 03): EVALUATION OF THE MODEL **14**
Evaluation 14
Fitness 14
Analysis 14
Example 15
Conclusion: 15
Precision 15
Analysis 16
Performance 16
Strengths:
The model captures the processes' behaviors. It replays some of the traces like:
Final Log (L) 16
} 16
Limitations: 16
● The model is unable to capture the relationship between the process B and F, and C and F. Hence it is underfitting 16

Introduction

In this assignment 2 and 3 of process mining, we have generated an event log from a process description and then applied 8 steps of alpha-algorithm on the event log. Furthermore, we have done the visualization of the petri net and analyzed its fitness and precision by evaluating the petri net.

Task 1 (ASSIGNMENT 02): EVENT LOG GENERATION PROGRAM

In task 1 of this assignment we implemented a python code that generates an event log from a given process description. This task takes the input in the form of process description and parameters that are taken into consideration to generate an event log.

Steps

1. When the code run, it asks the user to give certain inputs, these inputs are
    a. The number of traces.
    b. Amount of noise.
    c. Frequency of uncommon paths.
    d. Likelihood of missing events.
2. After these inputs the code reads the process description from the description.txt file, this is handled by the file_handeler.py file.
3. After taking in these inputs the code passes the input to Gemini Flash 1.5 via an api which processes the process description & parameters and returns event logs that are then displayed to the user.

```
Enter the number of traces: 20
Enter the amount of noise: 5
Enter the frequency of uncommon paths: 5
Enter the likelihood of missing events: 5
```

Task 2 (ASSIGNMENT 03): IMPLEMENTING THE ALPHA ALGORITHM

The event log is generated from the descriptions and an alpha algorithm is implemented on the event log which ultimately generates a petri net going through the following steps.

1. Step1: Display Final Log (L)

In this step, we have displayed Final log (L) which is a collection of unique traces generated from the event log as a response from API. Each trace is represented as a list and displayed with its frequency count.

```
STEP1:
--------------------------------------------------
Final Log (L)
--------------------------------------------------
{
    ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
    ['A', 'B', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
    ['A', 'C', 'B', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
    ['A', 'C', 'B', 'D', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
}
--------------------------------------------------
```

2. Step2: Extract and Sort Unique Events (TL), Initial Events (TI), and Final Events (TO)

In this step, we have extracted all the unique events from the traces and sorted them in alphabetical order.This gives us information about the starting and ending events in the process. Furthermore, we have categorized them as:

- **Unique Event (TL):** A list of all unique events from the event log
- **Initial Event (TI):** The first event in unique event list
- **Final Event (TO):** The last event in unique event list

```
------------------------------------------------
STEP2:
------------------------------------------------
Unique Events (TL): ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K']
Initial Events (TI): ['A']
Final Events (TO): ['K']
------------------------------------------------
```

3. Step 3: Determining Relationships (Causal, Parallel, Choice).

In this step, we have determined different relationships between events, which are further categorized as:

- **Causal Relationships:**

  This relationship occurs when x is directly followed by y but y is not directly followed by x.

  xy iff x>y and not y>x

- **Parallel Relationships:**

  This relationship occurs when x is directly followed by y but y is also directly followed by x.

  x||y iff x>y and y>x

- **Choice Relationships:**

  This relationship occurs when x is not directly followed by y but y is also not directly followed by x.

  x#y iff not x>y and not y>x

```
STEP3: Determining Relationships
--------------------------------------------------
Causal Relationships:
A -> B
A -> C
B -> D
C -> D
D -> E
D -> F
E -> F
F -> G
G -> H
H -> I
I -> J
J -> K
```

```
Parallel Relationships:
B || C
C || B
```

```
Choice Relationships:
A # D
A # E
A # F
A # G
A # H
A # I
A # J
A # K
B # E
B # F
B # G
B # H
B # I
B # J
B # K
C # E
C # F
```

4. Step 4: Footprint Matrix

In this step, we have made a footprint matrix table that visualizes causal, parallel and choice relationships. Each cell in the matrix shows the type of relationship between two events for instance, -> for causal, || for parallel and # for a choice relationship.

STEP4: Footprint Matrix

	A	B	C	D	E	F	G	H	I	J	K
A	#	->	->	#	#	#	#	#	#	#	#
B	<-	#	||	->	#	#	#	#	#	#	#
C	<-	||	#	->	#	#	#	#	#	#	#
D	#	<-	<-	#	->	->	#	#	#	#	#
E	#	#	#	<-	#	->	#	#	#	#	#
F	#	#	#	<-	<-	#	->	#	#	#	#
G	#	#	#	#	#	<-	#	->	#	#	#
H	#	#	#	#	#	#	<-	#	->	#	#
I	#	#	#	#	#	#	#	<-	#	->	#
J	#	#	#	#	#	#	#	#	<-	#	->
K	#	#	#	#	#	#	#	#	#	<-	#

5. Step 5: Generate Pair Sets of Causal Relationships (XL)

In this step, we have generated a list of pairs of events from causal relationships identified. The output shows a set of pairs representing direct causal relationships between events.

```
STEP5: Causal Pair Sets
--------------------------------------------------
({A, B})
({A, C})
({B, D})
({C, D})
({D, E})
({D, F})
({E, F})
({F, G})
({G, H})
({H, I})
({I, J})
({J, K})
--------------------------------------------------
```

6. Step 6: Displaying Maximal Relationship Pairs (YL)

In this step, we have generated maximal causal pair sets in which one event has multiple subsequent events. For instance if event A causes both event B and event C, we group them as one maximal pair: ({A}, {B,C})

```
STEP6: Maximal Relationship Pairs
-------------------------------------------------
({A}, {B, C})
({B}, {D})
({C}, {D})
({D}, {E, F})
({E}, {F})
({F}, {G})
({G}, {H})
({H}, {I})
({I}, {J})
({J}, {K})
-------------------------------------------------
```

7. Step 7: Place Set (PL) for Maximal Causal Pairs

In this step, we have generated place sets for maximal causal pairs. A place set is a representation of the system's state that includes all possible transitions or relationships between events. This step displays maximal causal pairs in the form of place sets.

```
STEP7: Place Set (PL) Maximal Causal Pairs
-------------------------------------------------
P({A}, {B, C})
P({B}, {D})
P({C}, {D})
P({D}, {E, F})
P({E}, {F})
P({F}, {G})
P({G}, {H})
P({H}, {I})
P({I}, {J})
P({J}, {K})
-------------------------------------------------
```

8. Step 8: Flow Relation (FL) for Maximal Causal Pairs

In this final step, we have generated flow relations for maximal causal pairs. It represents the flow of events based on their causal relationships. This helps in understanding the process's flow.

```
STEP8: Flow Relation for Maximal Causal Pairs
--------------------------------------------------
(A, P(A, { B, C }))
(P(A, { B, C }), B)
(P(A, { B, C }), C)
(B, P(B, { D }))
(P(B, { D }), D)
(C, P(C, { D }))
(P(C, { D }), D)
(D, P(D, { E, F }))
(P(D, { E, F }), E)
(P(D, { E, F }), F)
(E, P(E, { F }))
(P(E, { F }), F)
(F, P(F, { G }))
(P(F, { G }), G)
(G, P(G, { H }))
(P(G, { H }), H)
(H, P(H, { I }))
(P(H, { I }), I)
(I, P(I, { J }))
(P(I, { J }), J)
(J, P(J, { K }))
(P(J, { K }), K)
--------------------------------------------------
```

Task 3 (ASSIGNMENT 02): VISUALIZATION OF THE PROCESS MODEL

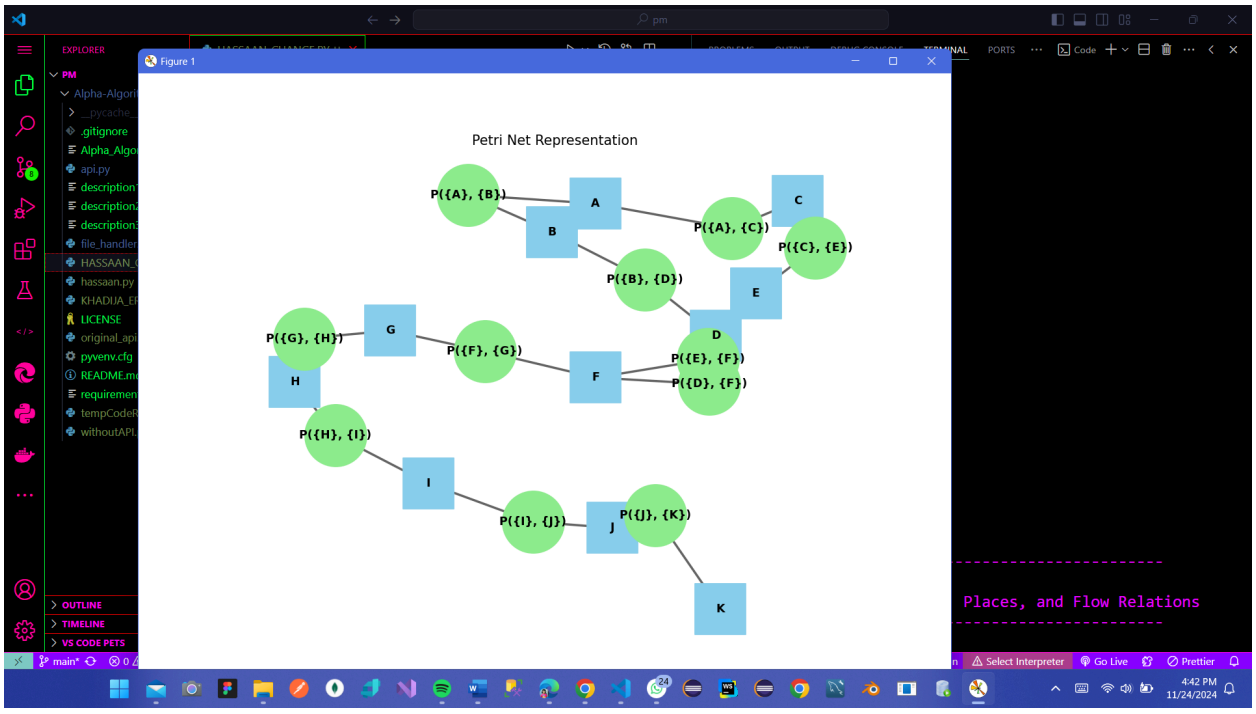
The next step involves the visualization of the process mode which includes the understanding of the process model and its structure by visualizing the petri net. It shows the connection of tasks, concurrent activities, choices and their sequences. The alpha algorithm is used to identify the transitions, places, and flow relations in the petri nets and arcs connecting places and transitions, showing the process flow.Initial place connects to the first task while final place connects from the last task.Hence ,the graphs shows the flow of activities in the process ,points of concurrency and starting and ending tasks.

```
Transitions:
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K']
```

```
Places:
P({A}, {B, C})
P({B}, {D})
P({C}, {D})
P({D}, {E, F})
P({E}, {F})
P({F}, {G})
P({G}, {H})
P({H}, {I})
P({I}, {J})
P({J}, {K})
```

```
Flow Relations:
(A, P(A, { B, C }))
(P(A, { B, C }), B)
(P(A, { B, C }), C)
(B, P(B, { D }))
(P(B, { D }), D)
(C, P(C, { D }))
(P(C, { D }), D)
(D, P(D, { E, F }))
(P(D, { E, F }), E)
(P(D, { E, F }), F)
(E, P(E, { F }))
(P(E, { F }), F)
(F, P(F, { G }))
(P(F, { G }), G)
(G, P(G, { H }))
(P(G, { H }), H)
(H, P(H, { I }))
(P(H, { I }), I)
(I, P(I, { J }))
(P(I, { J }), J)
(J, P(J, { K }))
(P(J, { K }), K)
----------------------------------------------------
```

Petri Net:



Task 4 (ASSIGNMENT 03): EVALUATION OF THE MODEL

Evaluation

Taking this event log as example let's evaluate the fitness and precision of our model:
Example Log:
Final Log (L)

{
['A', 'B', 'D', 'C', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
['A', 'B', 'D', 'E', 'C', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
['A', 'C', 'E', 'B', 'D', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
['A', 'C', 'E', 'D', 'B', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
}
Final Log (L)

{
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
['A', 'B', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
['A', 'C', 'B', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
['A', 'C', 'B', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
}

Fitness

Fitness measures how well the model can replay the event logs.

Analysis

All the traces in the event logs above are executed by the discovered petri net.
But from this event log:

Example

```
STEP1:
--------------------------------------------------
Final Log (L)
--------------------------------------------------
{
    ['A', 'B', 'D', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 10
    ['A', 'C', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 10
    ['A', 'B', 'D', 'F', 'G', 'H', 'I', 'J'], Frequency: 5
    ['A', 'C', 'E', 'F', 'G', 'H', 'I', 'J'], Frequency: 5
    ['A', 'B', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 2
    ['A', 'C', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 2
    ['A', 'B', 'F', 'G', 'H', 'I', 'J'], Frequency: 1
    ['A', 'C', 'F', 'G', 'H', 'I', 'J'], Frequency: 1
}
--------------------------------------------------
```

The log 2, log 3, log 4, log 5, log 6, log 7, log 8 cannot be executed. Indicating that the model is **underfit.**

Conclusion:

The model is underfit because it cannot replay many of the traces in the event log. The event logs which are frequently occurring in the event log are not executed by the model hence it is underfitting.

Precision

The model does not allow the traces to execute which are in event log making it underfit.For precision the underfitting is avoided. Hence the model is underfitting, it is less precise.

Analysis

Performance

Strengths:

The model captures the processes' behaviors. It replays some of the traces like:
Final Log (L)

{
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
['A', 'B', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
['A', 'C', 'B', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1
['A', 'C', 'B', 'E', 'F', 'G', 'H', 'I', 'J', 'K'], Frequency: 1

}

Limitations:

- The model is unable to capture the relationship between the process B and F, and C and F. Hence it is underfitting.
- The model is under fitting and less precise.