Process Mining: Using CPN Tools to Create Test Logs for Mining Algorithms

Ana Karla Alves de Medeiros and Christian W. Günther
Outline

• Introduction Process Mining
• The need for synthetic logs
• MXML format
• Using the Logging Extensions
• Conclusion
Process Mining - *The general idea*

![Diagram](image)

- Process Model
- Information System
- Logs
Process Mining - The general idea

- Basic Performance Metrics
- Process Model
- Auditing, Security
- Organizational Model
- Performance Characteristics
- Social Network
Process Mining - ... *what is it good for?*

• How is a specific process *really* executed?
  • Discover *discrepancies* process model ↔ reality
  • Discover typical *patterns* in real-life operation

• Results can be used to:
  • *Align* organization and processes in a better way
  • Improve *efficiency, effectivity* and *quality*
  • *Redesign* processes based on solid information
Process Mining - *Practice vs. Development*
Process Mining - *Practice vs. Development*

![Diagram of Process Mining](image-url)

- **PAIS**
- **Logs**
- **Process Mining Algorithm**
- **Comparison**
- **Process Model**
- **Mined Process Model**
Process Mining - Practice vs. Development

- Modeled informally
- Flexible process model
- Not available

Process Model

Comparison

- Incomplete
- Corrupt
- Noise

Mined Process Model

PAIS

- Problems during execution
- Logging mechanism flawed
- Aborted Instances
- Manual workarounds

Process Mining Algorithm

Logs
Process Mining - Practice vs. Development

- Modeled informally
- Flexible process model
- Not available

- Incomplete
- Corrupt
- Noise

- Problems during execution
- Logging mechanism flawed
- Aborted Instances
- Manual workarounds

Not suitable for testing mining algorithms!
Process Mining - *Practice vs. Development*

- **CPN Process Model**
- **Simulation in CPN Tools**
- **Logs**
- **Mined Process Model**
- **Process Mining Algorithm**
Process Mining - Practice vs. Development

CPN Process Model → Simulation in CPN Tools → Logs → Process Mining Algorithm → Mined Process Model

Comparison

- Complete
- Perfect
- Noise-free
Process Mining - Entirely controlled environment: Perfect testbed for Process Mining Algorithms!

Comparison

- Complete
- Perfect
- Noise-free

Simulation in CPN Tools

Logs

CPN Process Model

Mined Process Model

Process Mining Algorithm
Logging a CPN Simulation

• Example process model: Fine handling

```xml
<WorkflowLog>
  <Process>
    <ProcessInstance>
      <AuditTrailEntry/>
      <AuditTrailEntry/>
      <AuditTrailEntry/>
    </ProcessInstance>
    <ProcessInstance/>
    <ProcessInstance/>
  </Process>
</WorkflowLog>
```
The MXML Format - The Structure of an Audit Trail Entry

```xml
<AuditTrailEntry>
  <WorkflowModelElement/> Task A </WorkflowModelElement>
  <EventType> complete </EventType>
  <TimeStamp> 2005-10-26T12:37:33... </TimeStamp>
  <Originator> John Doe </Originator>
  <Data>
    <Attribute name="x"> 1 </Attribute>
    <Attribute name="y"> whatever </Attribute>
  </Data>
</AuditTrailEntry>
```
MXML Logging Extensions for CPN Tools

- Define 2 global constants
- Import ML file containing logging functions
- Call function `createCaseFile` to initialize new process instance
- Call function `addATE` to create new audit trail entry
HOWTO create an MXML log from a CPN Model:

• If not already done: *model the process in CPN Tools*
MXML Logging Extensions - *Control Variables*

- Three declarations are necessary:

```scala
- val FILE = "/logs/logCPN"
- val FILE_EXTENSION = ".cpnxm"
use "loggingFunctionsMultipleFiles.sml"
```

**FILE:** path and file prefix

**FILE_EXTENSION:** custom log file suffix

**import ML file** containing logging functions’ declarations
MXML Logging Extensions - Case File Creation

• function `createCaseFile`
  • one parameter `caseId` of type `Integer`
  • creates new file for recording a Process Instance
  • *File:* `<FILE> <caseId> <FILE_EXTENSION>`

• Example Use: **Case Generator**
  • extra transition at start of process
  • creates a defined number of tokens (=cases)
  • initializes log by calling `createCaseFile()`
MXML Logging Extensions - Case File Creation

Example application: Case Generator

Actual process starts here!
MXMLogging Extensions - Audit Trail Entries

• function **addATE**

  • Parameter list (**all MXML fields)**:
    • **caseId** (**Integer**): which **file** to append to
    • **workflowModelElement** (**String**): task name
    • **EventType** (**String**): complete, schedule, abort, ...
    • **TimeStamp** (**String**): when has the event occurred?
    • **Originator** (**String**): executing resource
    • **Data** (**List of Strings**): [name1, value1, name2, ...]

• Appends **audit trail entry** to case file with **caseId**
MXML Logging Extensions - Audit Trail Entries

- (convenience) function calculateTimeStamp
  - Takes no parameters
  - Creates MXML-compliant timestamp from current logical model time
**MXML Logging Extensions - Audit Trail Entries**

- **Usage of** *addATE*:

```xml
input (id, role1);
output ();
action
(addATE(id, "FileFine", ["complete"], calculateTimeStamp(), role1, []));
```

```xml
input (id, role0);
output ();
action
(addATE(id, "SendBill", ["complete"], calculateTimeStamp(), role0, []));
```
Aggregating Simulation Logs

- Combines all case files and writes clean **MXML**

1) Select CPN plugin

2) Select directory with logs

3) Provide file suffix

4) Run plugin and save MXML
Social Network Miner
Heuristics Miner
Multi-Phase Miner
Alpha algorithm
Conclusion

• Extending a model with Logging is straightforward
  • Combined with the ease of modeling in CPN Tools, rapid creation of test logs

• Intended Usage:
  • Synthesis of test logs for benchmarking mining algorithms
  • Creation of a test log repository
  • Testing mining algorithms on advanced models
    • “...could we mine your models?”