Process Mining (ProMi)

Winter 2015/16

Matthias Weidlich
I. The Context of Process Mining

Event logs and their creation
The Context
Logs as an Information Source

Logs contain information to answer questions
  • When have process instances been executed?
  • How many instances have been executed?
  • Have there been recurring patterns in the executions of activities?
  • Is it possible to construct process models based on the log data?
  • Which sequences of activities have been executed very frequently?
  • Does a process model contain execution sequences that have never been executed?

Logs are the basis for evidence-based answers to these questions
  • Not biased by human perception of how a process is conducted
  • Not biased by fragmentation of process knowledge
  • Yet, assuming high data quality
Log Entries

Example log entries

- Check of invoice with number 4567 finished on 12.11.2010 at 9:19:57


- Invoice sent for invoice number 4567 finished on 12.11.2010 at 9:23:18

- Inserted data (c1987, PromoMailing) into customer status database on 12.11.2010 at 9:24:10


- Check of invoice with number 4568 finished on 12.11.2010 at 9:26:38
Event Logs

Specific model that imposes assumptions
- (Total) order of events
- Event can be related to activity and process instance (case)
- Events in event log relate to instance of a single process

But
- Actual logs often contain events of various processes
- Requires pre-processing

Format of logs: (timestamp, caseID, activity, additional attributes ...)
- Example
  - *Check of invoice with number 4567 finished on 12.11.2010 at 9:19:57*

- Event log
  - *(1289553597, 4567, invoice-check),(1289553744, c1987, StoreCustomerData, “Müller”, “Bad Bentheim”)*
Event Log Abstractions

Further abstractions depending on use case
- Abstract from activity names (A’s and B’s)
- Abstract from specific timestamp values (preserve only ordering)
- Abstract from additional event data

Variations of event logs
- Not necessarily a 1:1 relation between activities and events
- Start and end events for execution of activities

Issues
- Quality issues in terms of missing events, duplicated events, incorrect ordering (due to race conditions), etc.
- Implications discussed later

| case 1 | task A |
| case 2 | task A |
| case 3 | task A |
| case 3 | task B |
| case 1 | task B |
| case 1 | task C |
| case 2 | task C |
| case 2 | task C |
| case 4 | task A |
| case 2 | task B |
| case 2 | task D |
| case 5 | task E |
| case 4 | task C |
| case 1 | task D |
| case 3 | task C |
| case 3 | task D |
| case 4 | task B |
| case 5 | task F |
| case 4 | task D |
Log Assumptions – Reality Check

Relate an event to the execution of an activity?
Relate an event to a process instance?

... Record claim
   Check coverage
     Request proof of loss
     Do field check
       Take decision...

... Submit order
   Check credit history
     Charge credit card
       Check availability
         Plan shipments...

Thinking further about:
Timestamps, snapshots, scoping, granularity
Extraction of Logs From DBs

Idea: Process-oriented Information Systems store process-related data in ordinary Databases
  • Changes in the data hint at activity execution
  • Logs about data changes enable construction of event logs

Issue: Relation between processes and data is non-trivial
  • To the rescue: Domain models outlining the entities can serve as a starting point
  • Approach allows for flexibility in the definition of the notion of a process instance

Details:
Eduardo González López de Murillas, Wil M. P. van der Aalst, Hajo A. Reijers: Process Mining on Databases: Unearthing Historical Data from Redo Logs. BPM 2015:367-385
DB Logging

1. runs a query to modify a row in a table
2. change is recorded in transaction log
3. query returns
4. change is recorded in data file

This occurs immediately when the change is made, and must complete before the query 'returns' to the user.

This happens during a checkpoint, and can be much later from when the change was originally made.
Transaction Log

Transaction Log (also Redo Log)

• Contains information on change operations
• Use to ensure consistent state of DB
  • Operations can be redone and undone
  • Operations are persisted
• Writing to transaction log is faster than writing to actual data file
  • Transaction log is written sequentially
  • Writing to data file: extension of data file, page splits, new allocations, etc.
Extraction Idea

Event extraction
  • Each record of transaction log is an event
  • Extract attributes affected by change operation

Exploiting the data model
  • Extract data model from DB
  • Rely on foreign keys in model to identify possible correlations

Case ID construction
  • Determine the scope to use for correlation
Simple scenario:
- Shop sells tickets
- Information between tickets and customers established via bookings
<table>
<thead>
<tr>
<th>#</th>
<th>Time</th>
<th>Op</th>
<th>Table</th>
<th>Redo</th>
<th>Undo</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2014-11-27</td>
<td></td>
<td>CUSTOMER</td>
<td>insert into &quot;SAMPLEDB&quot;. &quot;CUSTOMER&quot; (&quot;ID&quot;, &quot;NAME&quot;, &quot;ADDRESS&quot;, &quot;BIRTH_DATE&quot;) values ('17299', 'Name1', 'Address1', TO_DATE('01-AUG-06', 'DD-MON-RR'));</td>
<td>delete from &quot;SAMPLEDB&quot;. &quot;CUSTOMER&quot; where &quot;ID&quot; = '17299' and &quot;NAME&quot; = 'Name1' and &quot;ADDRESS&quot; = 'Address1' and &quot;BIRTH_DATE&quot; = TO_DATE('01-AUG-06', 'DD-MON-RR') and ROWID = '1';</td>
</tr>
<tr>
<td>2</td>
<td>2014-11-27</td>
<td></td>
<td>CUSTOMER</td>
<td>update &quot;SAMPLEDB&quot;. &quot;CUSTOMER&quot; set &quot;NAME&quot; = 'Name2' where &quot;NAME&quot; = 'Name1' and ROWID = '1';</td>
<td>update &quot;SAMPLEDB&quot;. &quot;CUSTOMER&quot; set &quot;NAME&quot; = 'Name2' and ROWID = '1';</td>
</tr>
<tr>
<td>3</td>
<td>2014-11-27</td>
<td></td>
<td>BOOKING</td>
<td>insert into &quot;SAMPLEDB&quot;. &quot;BOOKING&quot; (&quot;ID&quot;, &quot;CUSTOMER_ID&quot;) values ('36846', '17299');</td>
<td>delete from &quot;SAMPLEDB&quot;. &quot;BOOKING&quot; where &quot;ID&quot; = '36846' and &quot;CUSTOMER_ID&quot; = '17299' and ROWID = '2';</td>
</tr>
<tr>
<td>4</td>
<td>2014-11-27</td>
<td></td>
<td>TICKET</td>
<td>update &quot;SAMPLEDB&quot;. &quot;TICKET&quot; set &quot;BOOKING_ID&quot; = '36846' where &quot;BOOKING_ID&quot; IS NULL and ROWID = '3';</td>
<td>update &quot;SAMPLEDB&quot;. &quot;TICKET&quot; set &quot;BOOKING_ID&quot; = NULL where &quot;BOOKING_ID&quot; = '36846' and ROWID = '3';</td>
</tr>
<tr>
<td>5</td>
<td>2014-11-27</td>
<td></td>
<td>BOOKING</td>
<td>insert into &quot;SAMPLEDB&quot;. &quot;BOOKING&quot; (&quot;ID&quot;, &quot;CUSTOMER_ID&quot;) values ('36876', '17299');</td>
<td>delete from &quot;SAMPLEDB&quot;. &quot;BOOKING&quot; where &quot;ID&quot; = '36876' and &quot;CUSTOMER_ID&quot; = '17299' and ROWID = '4';</td>
</tr>
<tr>
<td>6</td>
<td>2014-11-27</td>
<td></td>
<td>TICKET</td>
<td>update &quot;SAMPLEDB&quot;. &quot;TICKET&quot; set &quot;ID&quot; = '36876' where &quot;BOOKING_ID&quot; IS NULL and ROWID = '5';</td>
<td>update &quot;SAMPLEDB&quot;. &quot;TICKET&quot; set &quot;ID&quot; = NULL where &quot;BOOKING_ID&quot; = '36876' and ROWID = '5';</td>
</tr>
</tbody>
</table>
## Extracted Events

<table>
<thead>
<tr>
<th>#</th>
<th>Attribute name</th>
<th>Value after event</th>
<th>Value before event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Customer:id</td>
<td>17299</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Customer:name</td>
<td>Name1</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Customer:address</td>
<td>Address1</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Customer:birth_date</td>
<td>01-AUG-06</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>2</td>
<td>Customer:id</td>
<td>=</td>
<td>{17299}</td>
</tr>
<tr>
<td></td>
<td>Customer:name</td>
<td>Name2</td>
<td>Name1</td>
</tr>
<tr>
<td></td>
<td>Customer:address</td>
<td>=</td>
<td>{Address1}</td>
</tr>
<tr>
<td></td>
<td>Customer:birth_date</td>
<td>=</td>
<td>{01-AUG-06}</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Booking:id</td>
<td>36846</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Booking:customer_id</td>
<td>17299</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Ticket:booking_id</td>
<td>36846</td>
<td>NULL</td>
</tr>
<tr>
<td></td>
<td>Ticket:id</td>
<td>=</td>
<td>(317132)</td>
</tr>
<tr>
<td></td>
<td>Ticket:belongs_to</td>
<td>=</td>
<td>(172935)</td>
</tr>
<tr>
<td></td>
<td>Ticket:for_concert</td>
<td>=</td>
<td>(1277)</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>Booking:id</td>
<td>36876</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Booking:customer_id</td>
<td>17299</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>Ticket:booking_id</td>
<td>36876</td>
<td>NULL</td>
</tr>
<tr>
<td></td>
<td>Ticket:id</td>
<td>=</td>
<td>(317435)</td>
</tr>
<tr>
<td></td>
<td>Ticket:belongs_to</td>
<td>=</td>
<td>(173238)</td>
</tr>
<tr>
<td></td>
<td>Ticket:for_concert</td>
<td>=</td>
<td>(1277)</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>5</td>
</tr>
</tbody>
</table>
Data Model Revisited

Foreign key relations hint at correlations between concepts!
## Possible Correlations

<table>
<thead>
<tr>
<th>#</th>
<th>Attribute name</th>
<th>Value after event</th>
<th>Value before event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Customer:id</td>
<td>17299</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Customer:name</td>
<td>Name1</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Customer:address</td>
<td>Address1</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Customer:birth_date</td>
<td>01-AUG-06</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Customer:id</td>
<td>=</td>
<td>{17299}</td>
</tr>
<tr>
<td></td>
<td>Customer:name</td>
<td>Name2</td>
<td>Name1</td>
</tr>
<tr>
<td></td>
<td>Customer:address</td>
<td>=</td>
<td>{Address1}</td>
</tr>
<tr>
<td></td>
<td>Customer:birth_date</td>
<td>=</td>
<td>{01-AUG-06}</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Booking:id</td>
<td>36846</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Booking:customer_id</td>
<td>17299</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Ticket:booking_id</td>
<td>36846</td>
<td>NULL</td>
</tr>
<tr>
<td></td>
<td>Ticket:id</td>
<td>=</td>
<td>(317132)</td>
</tr>
<tr>
<td></td>
<td>Ticket:belongs_to</td>
<td>=</td>
<td>(172935)</td>
</tr>
<tr>
<td></td>
<td>Ticket:for_concert</td>
<td>=</td>
<td>(1277)</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>Booking:id</td>
<td>36876</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Booking:customer_id</td>
<td>17299</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>Ticket:booking_id</td>
<td>36876</td>
<td>NULL</td>
</tr>
<tr>
<td></td>
<td>Ticket:id</td>
<td>=</td>
<td>(317435)</td>
</tr>
<tr>
<td></td>
<td>Ticket:belongs_to</td>
<td>=</td>
<td>(173238)</td>
</tr>
<tr>
<td></td>
<td>Ticket:for_concert</td>
<td>=</td>
<td>(1277)</td>
</tr>
<tr>
<td></td>
<td>RowID</td>
<td>=</td>
<td>5</td>
</tr>
</tbody>
</table>
Notions of a Case

Relations between foreign keys define “space” for case extraction

- All possible correlations
- Exact scoping depends on context

Here: case may be given by booking

- Split up dependency graph of events
- Extraction of two traces
Limitations

Strong link between CRUD operations and activities is assumed

- Record claim, Check coverage
- Submit order, Check credit history

Handling of operations related to multiple cases

Linking a case with primary keys in relations may be non-trivial
Next, different scenario:

- Relation between event types and activities is known
- The relation between events and cases is unknown
- There is no information in terms of a data model
- Assumption: log contains only the instance of a single process

Is there a way to separate cases in the log?
Exploit the fact that processes often follow sequential patterns

- Particular execution sequence is built of concatenations of these patterns
- Different instances of a process repeat these patterns

Estimate probabilistic model that is likely to explain the observed behaviour

Details:
Intuition
Intuition Cont.
Overview

Given recorded sequence $x$ of events, where event types (symbols) related to distinct activities

General approach

- Model:
  - Input sequence of symbols (activities) — $x$
  - Sequence of case IDs — $s$
  - Transition matrix over symbols — $M$
- Both, $s$ and $M$ are unknown
- Estimation based by Expectation-Maximization
  - Iterative method for finding maximum likelihood parameters
Transition Matrix

Underlying model is a Markow Chain

- Special symbols for start (○) and end (●)
- Matrix of conditional probabilities
  - $M(A,B)$ – probability of producing $B$ after $A$
  - $M(\cdot,\circ) = M(\bullet,\cdot) = M(\bullet,\circ) = 0$

\[
M = \begin{pmatrix}
  \circ & A & B & C & D & E & F & G & H & \bullet \\
  \circ & - & 1.0 & - & - & - & - & - & - & - \\
  A & - & - & 0.15 & 0.85 & - & - & - & - & - \\
  C & - & - & - & - & - & 0.47 & 0.53 & - & - \\
  E & - & - & - & - & - & - & - & - & 1.0 \\
  F & - & - & - & - & - & 0.5 & 0.5 & - & - \\
  H & - & - & - & - & - & - & - & - & 1.0 \\
  \bullet & - & - & - & - & - & - & - & - & - 
\end{pmatrix}
\]
Estimation of Transition Matrix

Estimate $M$ given $x$ and $s$

- Extract symbol sequences according to $s$
- Add start and end symbols

Joint probability of $x$ and $s$:

$$p(x, s) = \prod_{\text{symbol seq}} \prod_{\text{transitions}} M(\text{transition})$$

Estimator that maximises joint probability of $x$ and $s$, for pair of symbols $(a,b)$

$$\hat{M}(a, b) = \frac{\sum_{\text{symbol seq}} \# \text{ count}(a, b)}{\sum_{\text{symbol seq}} \sum_{\text{symbol}} \# \text{ count}(a, \text{symbol})}$$
Estimation Example

Symbol sequences

- 4x ○ACDEF●
- 9x ○ACDF●
- 4x ○ACDEGH●
- 3x ○AB●

Estimator for the transition from D to E

\[
\hat{M}(D, E) = \frac{4 \times 1 + 9 \times 0 + 4 \times 1 + 3 \times 0}{4 \times 1 + 9 \times 1 + 4 \times 1 + 3 \times 0} \approx 0.47
\]
Estimation of Sequence of Case IDs

Estimate s given x and M

- Find sequence of case IDs that maximises joint probability of x and s under given transition matrix
- Combinatorial optimisation problem

Estimation of s based on greedy algorithm

- Idea: if we know the previous symbol for each case, then the current symbol Z should be assigned to case that has the highest transition probability to produce Z

Basic steps:

- Select current symbol $x_i$ in input sequence x
- Determine which running case (not ending with $x_i$) has highest transition probability from its last symbol to $x_i$ and also consider $M(o, x_i)$ and $M(x_i, \circ)$
- Assign respective case ID to symbol $x_i$ or create new case (if $M(o, x_i)$ has max value) or close case (if $M(x_i, \circ)$ has max value)
- Move to next symbol in x
Estimation Example

At position 40 (G):
- 4 running cases (5,7,9,11) ending with E,G,D,A
- Assign to case ending with E

At position 41 (F):
- 4 cases ending with G,G,D,A
- Assign to case ending with D and close since M(F,●)

At position 42 (C):
- 3 cases (5,7,11) ending with G,G,A
- Assign to case ending with A

At position 43 (A):
- 3 cases ending with G,G,C
- Create new case
Overall Estimation Procedure

Estimate $M$ and $s$ given $x$

• Use initial estimate for $M$
• Proceed iteratively - Expectation Maximization (EM)
  • Expectation step:
    Use current matrix to estimate sequence of symbols
    (Estimate $s$ given $x$ and $M$)
  • Maximization step:
    Use symbol sequences to update transition matrix
    (Estimate $M$ given $x$ and $s$)
• Terminate once transition matrix does not change anymore

EM may get stuck in local maxima

• Random initialisation of $M$ is a bad idea
• Rather, $\hat{M}(a, b) = \frac{\# \text{count}(a,b)}{\sum_{\text{symbol}} \# \text{count}(a,\text{symbol})}$
Result of Estimation

Result is transition matrix – actually a process model (any issues?)

Result is also the sequence of case IDs – based thereon, various discovery algorithms can be applied
Limitations

Issues in the presence of concurrency

Particular problems with (short) loops

• Expectation step assumes that cases produce symbol at most once
• Dropping that assumption increases search space and chance of getting stuck in local maxima

Issues related to incomplete cases in input sequence (cases started earlier, or finishing later)
Quality of Event Logs

“Clean event logs“

• Event relates to one activity and one process instance
• All traces are valid execution sequences of the process

Not realistic in practice, there is “noise”

• Erroneous logging mechanisms
• Not all log entries are written, some are lost or inserted in a wrong order

Think about the example scenarios again....
Types of Noise

Logging was temporarily not available

Original Trace

| A | B | C | D | E | F | G | H | J | K | L | M | N |

Noisy Trace

| A | B | C | D | E | F | G | H | J |

Missing Tail

Original Trace

| A | B | C | D | E | F | G | H | J | K | L | M | N |

Noisy Trace

| E | F | G | H | J | K | L | M | N |

Missing Head

Original Trace

| A | B | C | D | E | F | G | H | J | K | L | M | N |

Noisy Trace

| A | B | C | D |

Missing Episode

| K | L | M | N |
Types of Noise cont.

Original Trace

A B C D E F G H J K L M N

Noisy Trace

A B C K E F G H J D L M N

Perturbed Order

Original Trace

A B C D E F G H J K L M

Noisy Trace

A B C B D E F G H J K L M

Additional Event

Original Trace

A B C D E F G H J K L M

Noisy Trace

A B C X D E F G H J K L M

Alien Event
Consequences of Noise

Massive impact on discovery, conformance, and enhancement techniques – we will get back to this

Already an issue in the construction of event logs

Major issue: *what* is noise is close to impossible to characterise without domain knowledge
Noise Example

ACD 99
ACE 0
BCE 85
BCD 0
Noise Example cont.

ACD 99
ACE 88
BCE 85
BCD 78
Example cont.

ACD 99
ACE 2
BCE 85
BCD 3
Log-based Noise Handling

Rely on frequency analysis to identify noise in event log

Assumption: noise is rare
  • Very infrequent traces can be seen as noise
  • Traces that contain very infrequent transitions can be seen as noise
  • Operationalisation based on standard data mining techniques – association rules mining

Again, this assumption may be wrong!
Association rules

Apriori algorithm for frequent itemset mining

Subsequent construction of association rules by investigating support of subsets
Mine Rules of Traces

Adopt the same association rules approach:

• Instead of sets of items, use sequences of items
• Instead of unifying sets, concatenate sequences

Results:

• Identifies frequent traces
• Identifies frequent trace continuations
• Thresholding to characterize noise
Practical Considerations

Event logs take various different forms and instantiations

Differences in semantics, e.g., related to
- Timestamps
- Total vs. Partial order

Difference in quality, e.g., related to
- Completeness
- Noise-level
- Data richness

Technical alignment by means of standards
But: Semantic alignment a major issue
Take Away

Event logs as the basis of process mining techniques

Essential assumptions on event ordering and identifiers for activities and cases

Construction of event log may be supported by domain models or probabilistic techniques