Process Mining: Beyond Business Intelligence

prof.dr.ir. Wil van der Aalst
www.processmining.org
Today's information systems are really crappy compared to a TomTom system!

- Good maps?
- Navigation by PowerPoints?
- Traffic information?
- Where is the next fuel station?
- Who is in charge?
- Seamless zoom?
- Customizable views?
- When will the destination be reached?
State of the art in process modeling?

Models are not accurate enough!

Not of TomTom quality!

Process mining allows for better maps, better navigation, better traffic information, etc.
Process Mining

- Process discovery: "What is really happening?"
- Conformance checking: "Do we do what was agreed upon?"
- Performance analysis: "Where are the bottlenecks?"
- Process prediction: "Will this case be late?"
- Process improvement: "How to redesign this process?"
- Etc.
- Process discovery: "What is the real curriculum?"
- Conformance checking: "Do students meet the prerequisites?"
- Performance analysis: "Where are the bottlenecks?"
- Process prediction: "Will a student complete his studies (in time)?"
- Process improvement: "How to redesign the curriculum?"
Outline of tutorial

• Part I : Introduction to Process Mining
• Part II : Process Discovery – The Alpha Algorithm
• Part III : Hands-on with ProM
Part I
Introduction to Process Mining

www.processmining.org
Where to start?

process design

implementation/configuration

process enactment

process control

process diagnosis

process mining
Role of models

"world"
- business processes
- people
- machines
- components
- organizations

supports/controls

software system

specifies configures implements analyzes

"real world"

"powerpoint reality"
Event logs are a reflection of reality

“world”
- business processes
- people
- machines
- components
- organizations

supports/controls

software system

records events, e.g., messages, transactions, etc.

event logs
Examples:
Process mining: Linking events to models
Starting point: event logs

event logs, audit trails, databases, message logs, etc.

unified event log (MXML)
Discovery

```
“world”
business processes
people
machines
components
organizations
models
analyzes

software system
specifies
configures
implements
analyzes

process/system
model

supports/controls

records
events, e.g.,
messages,
transactions,
etc.

event logs

conformance
discovery
```
What to discover?

- process models (Petri nets, EPCs, BPMN, etc.),
- organizational models,
- social networks,
- sequence diagrams,
- business rules,
- bottlenecks,
- simulation models,
- etc.

i.e., beyond "slice and dice" and showing KPIs on a dashboard ...
MXML Log
- instances: 3512
- audit trail entries: 46138

ProM supports +40 types of model discovery!
bottlenecks

flow time from A to B

throughput time
46138 events

short cases

time (relative)

long cases
A bit of theory: Process discovery techniques

- Algorithmic techniques
  - Alpha miner
  - Alpha+, Alpha++, Alpha#
  - Heuristic miner
  - Multi phase miner
  - ...
- Genetic process mining
- Region-based process mining
  - State-based regions
  - Language based regions

cf. www.processmining.org for an overview
Example: Genetic Mining

1. initial population
2. fitness test
3. select best parents
4. crossover
5. children
6. mutation
7. new population

used in e.g. ProM, Futura Reflex, BPM|one
Conformance Checking

“world”
- business processes
- people
- machines
- components
- organizations

models
- analyzes

process/
- system
model

software
system

supports/
controls

specifies
configures
implements
analyzes

records
- events, e.g.,
- messages,
- transactions,
- etc.

discovery

conformance

event
logs
Conformance Checking

• Compare process model and event log: highlight deviations and measure conformance.
• Compare constraints/business rules and event logs: check e.g. the 4-eyes principle.
Process mining as a mirror ...
Tool support
• Open source initiative started in 2003 after several early prototypes.
• Common Public License (CPL).
• Current version: 5.0 (5.2)
• ProMimport: to extract MXML from all kinds of applications
• Plug-in architecture.
• About 250 plug-ins available:
  • mining plug-ins: 38 (all mining algorithms presented and many more)
  • analysis plug-ins: 71 (e.g., verification, SNA, LTL, conformance checking, etc.)
  • import: 21 (for loading EPCs, Petri nets, YAWL, BPMN, etc.)
  • export: 44 (for storing EPCs, Petri nets, YAWL, BPMN, BPEL, etc.)
  • conversion: 45 (e.g., translating EPCs or BPMN into Petri nets)
  • filter: 24 (e.g., removing infrequent activities)
Business Intelligence Tools?

- Business Objects (SAP)
- Cognos Business Intelligence (IBM)
- Oracle Business Intelligence
- Hyperion (Oracle)
- SAS Business Intelligence
- Microsoft Business Intelligence
- SAP Business Intelligence (SAP BI)
- Jaspersoft (Open Source Business Intelligence)
- Pentaho BI Suite (Open Source)
- 
- Dashboards, reports, scorecards, ...
- Slicing and dicing, data mining, ...
Process Mining: Applications
Where did we apply process mining?

- Municipalities (e.g., Alkmaar, Heusden, Harderwijk, etc.)
- Government agencies (e.g., Rijkswaterstaat, Centraal Justitieel Incasso Bureau, Justice department)
- Insurance related agencies (e.g., UWV)
- Banks (e.g., ING Bank)
- Hospitals (e.g., AMC hospital, Catharina hospital)
- Multinationals (e.g., DSM, Deloitte)
- High-tech system manufacturers and their customers (e.g., Philips Healthcare, ASML, Thales)
- Media companies (e.g. Winkwaves)
- ...
Example: WMO process of a Dutch Municipality

WMO = Wet Maatschappelijke Ondersteuning

144 cases
1326 events
Conformance check of discovered model

- Conformance check of discovered model
- Activity is sometimes not performed
- Good fit 97.9%
- Drill down performed while not allowed
- Activity is sometimes not performed
- Both
Events sorted by start time of case
Events sorted by duration
## Idle time versus working time

<table>
<thead>
<tr>
<th>Component type:</th>
<th>Instance ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time option:</td>
<td>Actual</td>
</tr>
<tr>
<td>Relative Time option:</td>
<td>Instance ID</td>
</tr>
<tr>
<td>Time sort (chart):</td>
<td>months</td>
</tr>
<tr>
<td>Mouse Mode:</td>
<td>Zoom in</td>
</tr>
<tr>
<td>Zoom in:</td>
<td>(10^3)x</td>
</tr>
<tr>
<td>zoom (X)</td>
<td>(10^2)x</td>
</tr>
<tr>
<td>zoom (Y)</td>
<td>10x</td>
</tr>
<tr>
<td>1x</td>
<td>(10^2)x</td>
</tr>
<tr>
<td>zoom (X)</td>
<td>1x</td>
</tr>
<tr>
<td>zoom (Y)</td>
<td>(10^3)x</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element State Chart</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.3.2008</td>
<td>9.4.2008</td>
</tr>
<tr>
<td>5.5.2008</td>
<td>6.6.2008</td>
</tr>
<tr>
<td>8.7.2008</td>
<td>7.8.2008</td>
</tr>
<tr>
<td>5.9.2008</td>
<td>6.10.2008</td>
</tr>
</tbody>
</table>

- Buffer log reader created from reader BufferedLogReader: 144 Process instances and 1326 Audit Trail Entries from "D:\application\data\PromPallasWMO.Har"
"Real" animation
And of course ...
Reality ≠ PowerPoint (or Visio)
Process spectrum

structured (Lasagna)

unstructured (Spaghetti)
375 houses
18640 events
82 different activities
2712 patients
29258 events
264 different activities
874 patients
10478 events
181 different activities
24 machines
154966 events
360 different activities
Process Mining: TomTom for Business Processes
How can process mining help?

- Good maps?
- Navigation by PowerPoints?
- Traffic information?
- Where is the next fuel station?
- Who is in charge?
- Seamless zoom?
- Customizable views?
- When will the destination be reached?
ProM's "real animation"
When will I be home?
Approach

When?

- Operational business processes
- Information system
- Event log
- Process
- Prediction engine
- Annotated transition system

12-6-2009!
Input: partial trace and historic information

(A B C D C D C D E)?

(12-6-2009)!

(14-6-2009)!
### Input

<table>
<thead>
<tr>
<th>event id</th>
<th>timestamp</th>
<th>activity</th>
<th>resource</th>
<th>cost</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>35654423</td>
<td>30-12-2008:11.10</td>
<td>A</td>
<td>John</td>
<td>300</td>
<td>...</td>
</tr>
<tr>
<td>35654424</td>
<td>30-12-2008:15.21</td>
<td>B</td>
<td>John</td>
<td>400</td>
<td>...</td>
</tr>
<tr>
<td>35654425</td>
<td>30-12-2008:15.35</td>
<td>C</td>
<td>John</td>
<td>100</td>
<td>...</td>
</tr>
<tr>
<td>35654426</td>
<td>30-12-2008:15.55</td>
<td>D</td>
<td>John</td>
<td>400</td>
<td>...</td>
</tr>
<tr>
<td>35655526</td>
<td>29-12-2008:16.15</td>
<td>A</td>
<td>Ann</td>
<td>300</td>
<td>...</td>
</tr>
<tr>
<td>35655527</td>
<td>30-12-2008:16.05</td>
<td>C</td>
<td></td>
<td>10</td>
<td>...</td>
</tr>
<tr>
<td>35655528</td>
<td>30-12-2008:16.25</td>
<td>B</td>
<td></td>
<td>20</td>
<td>...</td>
</tr>
<tr>
<td>35655529</td>
<td>31-12-2008:10.55</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

...     ...     ...     ...     ...

<table>
<thead>
<tr>
<th></th>
<th>( \langle A^{00}, B^{06}, C^{12}, D^{18} \rangle )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>( \langle A^{10}, C^{14}, B^{26}, D^{36} \rangle )</td>
</tr>
<tr>
<td>3</td>
<td>( \langle A^{12}, E^{22}, D^{56} \rangle )</td>
</tr>
<tr>
<td>4</td>
<td>( \langle A^{15}, B^{19}, C^{22}, D^{28} \rangle )</td>
</tr>
<tr>
<td>5</td>
<td>( \langle A^{18}, B^{22}, C^{26}, D^{32} \rangle )</td>
</tr>
<tr>
<td>6</td>
<td>( \langle A^{19}, E^{28}, D^{59} \rangle )</td>
</tr>
<tr>
<td>7</td>
<td>( \langle A^{20}, C^{25}, B^{36}, D^{44} \rangle )</td>
</tr>
</tbody>
</table>
Building transition systems

(a) transition system based on sets

(b) transition system based on sequences

many abstractions are possible and supported by ProM's FSM miner
Annotated transition system based on remaining time

<table>
<thead>
<tr>
<th></th>
<th>( \langle \text{A}^{00}, \text{B}^{06}, \text{C}^{12}, \text{D}^{18} \rangle )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>( \langle \text{A}^{10}, \text{C}^{14}, \text{B}^{26}, \text{D}^{36} \rangle )</td>
</tr>
<tr>
<td>3</td>
<td>( \langle \text{A}^{12}, \text{E}^{22}, \text{D}^{56} \rangle )</td>
</tr>
<tr>
<td>4</td>
<td>( \langle \text{A}^{15}, \text{B}^{19}, \text{C}^{22}, \text{D}^{28} \rangle )</td>
</tr>
<tr>
<td>5</td>
<td>( \langle \text{A}^{18}, \text{B}^{22}, \text{C}^{26}, \text{D}^{32} \rangle )</td>
</tr>
<tr>
<td>6</td>
<td>( \langle \text{A}^{19}, \text{E}^{28}, \text{D}^{59} \rangle )</td>
</tr>
<tr>
<td>7</td>
<td>( \langle \text{A}^{20}, \text{C}^{25}, \text{B}^{36}, \text{D}^{14} \rangle )</td>
</tr>
</tbody>
</table>

\( [12,9,10] \)  \( [6,10,6,6,8] \)  \( [0,0,0,0,0] \)

\( [18,26,44,13, 14,40,24] \)  \( [22,19] \)  \( [34,31] \)  \( [0,0] \)
Predictive information

average: 10.33
st. dev.: 1.53
min: 9
max: 12

average: 7.2
st. dev.: 1.79
min: 6
max: 10

average: 0
st. dev.: 0
min: 0
max: 0

average: 25.75
st. dev.: 12.25
min: 13
max: 44

average: 25.75
st. dev.: 12.25
min: 13
max: 44

average: 32.5
st. dev.: 2.12
min: 31
max: 34

average: 20.5
st. dev.: 2.12
min: 19
max: 22

average: 0
st. dev.: 0
min: 0
max: 0

predict: 10.33
[12,9,10]

predict: 7.2
[6,10,6,6,8]

predict: 25.75
[18,26,44,13,14,40,24]

predict: 0
[0,0,0,0,0]

predict: 20.5
[34,31]

predict: 0
[0,0]
Example: WOZ process in Dutch Municipality

1882 objections triggering 11985 activities
All 11985 events at a glance

Average flow time is 107 days (with a huge variation)
For partial traces corresponding to this state the estimated time until completion is 8.5 days.
Cross validation: training set and test set

- Mean Average Error (MAE)
- Rooted MSE
- MAPE
Some results

(a) Set abstraction based on all activities

<table>
<thead>
<tr>
<th>Abstraction</th>
<th>MAE</th>
<th>RMSE</th>
<th>MAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set abstraction based on all activities (cf. Figure 17(a))</td>
<td>41.648</td>
<td>47.513</td>
<td>1505.07</td>
</tr>
<tr>
<td>Set abstraction based on last activity (cf. Figure 17(b))</td>
<td>43.080</td>
<td>49.666</td>
<td>1818.49</td>
</tr>
<tr>
<td>Set abstraction based on last activity and additional information related to the occurrence of “I” (cf. Figure 17(c))</td>
<td>17.129</td>
<td>23.550</td>
<td>900.07</td>
</tr>
<tr>
<td>Complete abstraction (cf. Figure 17(d))</td>
<td>63.391</td>
<td>74.965</td>
<td>7169.55</td>
</tr>
<tr>
<td>Simple heuristic: half of average total flow time (53.57 days)</td>
<td>61.750</td>
<td>75.505</td>
<td>6188.04</td>
</tr>
</tbody>
</table>
Part II
Process Discovery – The Alpha Algorithm

www.processmining.org
Design-time analysis vs run-time analysis

- Design-time analysis
  - Models
    - process model
    - (software) system
  - Analyzes
  - Verifies
  - Validates
  - Performs analysis
  - Supports/controls

- Run-time analysis
  - Records events, e.g., messages, transactions, etc.
  - Event logs
  - Discovery
  - Conformance
  - Extension

- Models
  - Business processes
  - People
  - Services
  - Components
  - Organizations

- Systems
  - e.g., systems like WebSphere, Oracle, TIBCO/Staffware, SAP, FLOWer, etc.

- Event logs
  - e.g., dedicated formats such as IBM’s Common Event Infrastructure (CEI) and MXML or proprietary formats stored in flat files or database tables.
Starting point: event logs

event logs, audit trails, databases, message logs, etc.

unified event log (MXML)
Event log:
- processes
- process instances
  - events

Per event:
- activity name
- (event type)
- (originator)
- (timestamp)
- (data)
<ProcessInstance>
  - <AuditTrailEntry>
    - <WorkflowModelElement>invite reviewers</WorkflowModelElement>
    - <EventType>start</EventType>
    - <Timestamp>2006-08-28T00:00:00.000+01:00</Timestamp>
    - <Originator>Mike</Originator>
  </AuditTrailEntry>
  - <AuditTrailEntry>
    - <WorkflowModelElement>invite reviewers</WorkflowModelElement>
    - <EventType>complete</EventType>
    - <Timestamp>2006-08-31T00:00:00.000+01:00</Timestamp>
    - <Originator>Mike</Originator>
  </AuditTrailEntry>
  - <AuditTrailEntry>
    - <Data>
      - <Attribute name="result">reject</Attribute>
    </Data>
    - <WorkflowModelElement>get review 3</WorkflowModelElement>
    - <EventType>complete</EventType>
    - <Timestamp>2006-09-02T00:00:00.000+01:00</Timestamp>
    - <Originator>Mary</Originator>
  </AuditTrailEntry>
  - <AuditTrailEntry>
    - <WorkflowModelElement>time out 1</WorkflowModelElement>
    - <EventType>complete</EventType>
    - <Timestamp>2006-09-03T00:00:00.000+01:00</Timestamp>
    - <Originator/>
  </AuditTrailEntry>
  - <AuditTrailEntry>
    - <WorkflowModelElement>time out 2</WorkflowModelElement>
    - <EventType>complete</EventType>
    - <Timestamp>2006-09-04T00:00:00.000+01:00</Timestamp>
    - <Originator/>
  </AuditTrailEntry>
</ProcessInstance>
Process Mining: The alpha algorithm
Alpha algorithm
Without transactional information (just completes)
Example log

- Minimal information in log: case id’s and task id’s.
- Additional information: event type, time, resources, and data.
- Sequences:
  - 1: ABCD
  - 2: ACBD
  - 3: ABCD
  - 4: ACBD
  - 5: EF
- So this log there are three possible sequences:
  - ABCD
  - ACBD
  - EF

| 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 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 |
> , → , || , # relations

- **Direct succession:** $x > y$ iff for some case $x$ is directly followed by $y$.
- **Causality:** $x \rightarrow y$ iff $x > y$ and not $y > x$.
- **Parallel:** $x || y$ iff $x > y$ and $y > x$.
- **Choice:** $x \# y$ iff not $x > y$ and not $y > x$.

```
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 4 : task A
case 2 : task B
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 1 : task D
case 3 : task C
case 3 : task D
case 4 : task B
```

Relations:
- $A > B$
- $A > C$
- $B > C$
- $B > D$
- $C > B$
- $C > D$
- $E > F$
- $A \rightarrow B$
- $A \rightarrow C$
- $B \rightarrow D$
- $C \rightarrow D$
- $E \rightarrow F$
- $B || C$
- $C || B$
Basic idea (1)

\[ x \rightarrow y \]
Basic idea (2)

\[ x \rightarrow y, \ x \rightarrow z, \ \text{and} \ y \parallel z \]
Basic idea (3)

\[ x \rightarrow y, \ x \rightarrow z, \ \text{and} \ y \# z \]
Basic idea (4)

$x \rightarrow z$, $y \rightarrow z$, and $x \parallel y$
Basic idea (5)

\[ x \rightarrow z, \ y \rightarrow z, \ \text{and} \ x \# y \]
It is not that simple!
Basic alpha algorithm

Let $W$ be a workflow log over $T$. $\alpha(W)$ is defined as follows.

1. $T_W = \{ t \in T \mid \exists \sigma \in W \ t \in \sigma \}$,
2. $T_I = \{ t \in T \mid \exists \sigma \in W \ t = first(\sigma) \}$,
3. $T_O = \{ t \in T \mid \exists \sigma \in W \ t = last(\sigma) \}$,
4. $X_W = \{ (A,B) \mid A \subseteq T_W \land A \neq \emptyset \land B \subseteq T_W \land B \neq \emptyset \land \forall a \in A \ \forall b \in B \ a \rightarrow_W b \land \forall a_1,a_2 \in A \ a_1 \#_W a_2 \land \forall b_1,b_2 \in B \ b_1 \#_W b_2 \}$,
5. $Y_W = \{ (A,B) \in X \mid \forall (A',B') \in X \ A \subseteq A' \land B \subseteq B' \Rightarrow (A,B) = (A',B') \}$,
6. $P_W = \{ p_{(A,B)} \mid (A,B) \in Y_W \} \cup \{ i_W, o_W \}$,
7. $F_W = \{ (a,p_{(A,B)}) \mid (A,B) \in Y_W \land a \in A \} \cup \{ (p_{(A,B)},b) \mid (A,B) \in Y_W \land b \in B \} \cup \{ (i_W,t) \mid t \in T_I \} \cup \{ (t,o_W) \mid t \in T_O \}$, and
8. $\alpha(W) = (P_W,T_W,F_W)$. 
Example revisited

$W$:  
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 A  
case 1: task D  
case 3: task C  
case 3: task D  
case 4: task B  
case 5: task F  
case 4: task D

$\alpha(W)$

A→B  
A→C  
B→D  
C→D  
E→F

B||C  
C||B
Exercise (1)

- What does the alpha algorithm produce for a log consisting only of the following traces?
  - ABCD
  - ACBD
  - AED

Let $W$ be a workflow log over $T$. $\alpha(W)$ is defined as follows.
1. $T_W = \{ t \in T \mid \exists \sigma \in W \ t \in \sigma \}$,
2. $T_I = \{ t \in T \mid \exists \sigma \in W \ t = \text{first}(\sigma) \}$,
3. $T_O = \{ t \in T \mid \exists \sigma \in W \ t = \text{last}(\sigma) \}$,
4. $X_W = \{ (A,B) \mid A \subseteq T_W \land A \neq \emptyset \land B \subseteq T_W \land B \neq \emptyset \land \forall a \in A \forall b \in B \ a \rightarrow_W b \land \forall a_1, a_2 \in A \ a_1 \#_W a_2 \land \forall b_1, b_2 \in B \ b_1 \#_W b_2 \}$,
5. $Y_W = \{ (A,B) \in X \mid \forall (A',B') \in X A \subseteq A' \land B \subseteq B' \Rightarrow (A,B) = (A',B') \}$,
6. $P_W = \{ p_{(A,B)} \mid (A,B) \in Y_W \} \cup \{i_W, o_W\}$,
7. $F_W = \{ (a,p_{(A,B)}) \mid (A,B) \in Y_W \land a \in A \} \cup \{ (p_{(A,B)}, b) \mid (A,B) \in Y_W \land b \in B \} \cup \{ (i_W, t) \mid t \in T_I \} \cup \{ (t, o_W) \mid t \in T_O \}$, and
8. $\alpha(W) = (P_W, T_W, F_W)$. 
Definition 3.1 (Workflow trace, Workflow log). Let $T$ be a set of tasks. $\sigma \in T^*$ is a workflow trace and $W \in \mathcal{P}(T^*)$ is a workflow log.²

The workflow trace of case 1 in Table 1 is $ABCD$. The workflow log corresponding to Table 1 is 

$\{ABCD, ACBD, AED\}$.

<table>
<thead>
<tr>
<th>case identifier</th>
<th>task identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>case 1</td>
<td>task A</td>
</tr>
<tr>
<td>case 2</td>
<td>task A</td>
</tr>
<tr>
<td>case 3</td>
<td>task A</td>
</tr>
<tr>
<td>case 1</td>
<td>task B</td>
</tr>
<tr>
<td>case 3</td>
<td>task B</td>
</tr>
<tr>
<td>case 1</td>
<td>task C</td>
</tr>
<tr>
<td>case 2</td>
<td>task C</td>
</tr>
<tr>
<td>case 4</td>
<td>task A</td>
</tr>
<tr>
<td>case 2</td>
<td>task B</td>
</tr>
<tr>
<td>case 2</td>
<td>task D</td>
</tr>
<tr>
<td>case 5</td>
<td>task A</td>
</tr>
<tr>
<td>case 4</td>
<td>task C</td>
</tr>
<tr>
<td>case 1</td>
<td>task D</td>
</tr>
<tr>
<td>case 3</td>
<td>task C</td>
</tr>
<tr>
<td>case 3</td>
<td>task D</td>
</tr>
<tr>
<td>case 4</td>
<td>task B</td>
</tr>
<tr>
<td>case 5</td>
<td>task E</td>
</tr>
<tr>
<td>case 5</td>
<td>task D</td>
</tr>
<tr>
<td>case 4</td>
<td>task D</td>
</tr>
</tbody>
</table>

Taken from:
Definition 3.2 (Log-based ordering relations). Let $W$ be a workflow log over $T$, i.e., $W \in \mathcal{P}(T^*)$. Let $a, b \in T$:

- $a >_W b$ iff there is a trace $\sigma = t_1t_2t_3 \ldots t_{n-1}$ and $i \in \{1, \ldots, n-2\}$ such that $\sigma \in W$ and $t_i = a$ and $t_{i+1} = b$,
- $a \rightarrow_W b$ iff $a >_W b$ and $b \not>_W a$,
- $a \#_W b$ iff $a \not>_W b$ and $b \not>_W a$, and
- $a ||_W b$ iff $a >_W b$ and $b >_W a$.

Consider the workflow log $W = \{ABCD, ACBD, AED\}$ (i.e., the log shown in Table 1). Relation $>_W$ describes which tasks appeared in sequence (one directly following the other). Clearly, $A >_W B$, $A >_W C$, $A >_W E$, $B >_W C$, $B >_W D$, $C >_W B$, $C >_W D$, and $E >_W D$. Relation $\rightarrow_W$ can be computed from $>_W$ and is referred to as the (direct) causal relation derived from workflow log $W$. $A \rightarrow_W B$, $A \rightarrow_W C$, $A \rightarrow_W E$, $B \rightarrow_W D$, $C \rightarrow_W D$, and $E \rightarrow_W D$. Note that $B \not\rightarrow_W C$ because $C >_W B$. Relation $||_W$ suggests potential parallelism. For log $W$, tasks $B$ and $C$ seem to be in parallel, i.e., $B||_W C$ and $C||_W B$. If two tasks can follow each other directly in any order, then all possible interleavings are present and,
Definition 4.3 (Mining algorithm $\alpha$). Let $W$ be a workflow log over $T$. $\alpha(W)$ is defined as follows:

1. $T_W = \{ t \in T \mid \exists \sigma \in W t \in \sigma \}$,
2. $T_I = \{ t \in T \mid \exists \sigma \in W t = first(\sigma) \}$,
3. $T_O = \{ t \in T \mid \exists \sigma \in W t = last(\sigma) \}$,

4. $X_W = \{(A, B) \mid \forall a \in A \forall \sigma \in W (t \in \sigma) \Rightarrow t = a \}$,

5. $Y_W = \{(A, B) \mid B \subseteq B' \Rightarrow (A, B) = (A', B') \}$,

6. $P_W = \{ p_{(A,B)} \mid (A, B) \in Y_W \} \cup \{i_W, o_W \}$,

7. $F_W = \{ (a, p_{(A,B)}) \mid (A, B) \in Y_W \land a \in A \}
\cup \{ (p_{(A,B)}, b) \mid (A, B) \in Y_W \land b \in B \}
\cup \{ (i_W, t) \mid t \in T_I \} \cup \{ (t, o_W) \mid t \in T_O \}$,

and

8. $\alpha(W) = (P_W, T_W, F_W)$.
Definition 4.3 (Mining algorithm $\alpha$). Let $W$ be a workflow log over $T$. $\alpha(W)$ is defined as follows:

1. $T_W = \{ t \in T \mid \exists \sigma \in W t \in \sigma \}$,
2. $T_I = \{ t \in T \mid \exists \sigma \in W t = \text{first}(\sigma) \}$,
3. $T_O = \{ t \in T \mid \exists \sigma \in W t = \text{last}(\sigma) \}$,
4. $X_W = \{ (A, B) \mid A \subseteq T_W \wedge B \subseteq T_W \wedge \forall a \in A \forall b \in B a \rightarrow_W b \wedge \forall a_1, a_2 \in A a_1 \#_W a_2 \wedge \forall b_1, b_2 \in B b_1 \#_W b_2 \}$,
5. $X_W = \{ (\{A\}, \{B\}), (\{A\}, \{C\}), (\{A\}, \{E\}), (\{B\}, \{D\}), (\{C\}, \{D\}), (\{E\}, \{D\}), (\{A\}, \{B, E\}), (\{A\}, \{C, E\}), (\{B, E\}, \{D\}), (\{C, E\}, \{D\}) \}$,
4. 

\[ X_W = \{ (\{A\}, \{B\}), (\{A\}, \{C\}), (\{A\}, \{E\}), (\{B\}, \{D\}), (\{C\}, \{D\}), (\{E\}, \{D\}), (\{A\}, \{B, E\}), (\{A\}, \{C, E\}), (\{B, E\}, \{D\}), (\{C, E\}, \{D\}) \}, \]

\[ W = \{ ABCD, ACBD, AED \} \]

5. 

\[ Y_W = \{ (A, B) \in X_W \mid \forall (A', B') \in X_W A \subseteq A' \]
\[ \wedge B \subseteq B' \Rightarrow (A, B) = (A', B') \} \]

\[ Y_W = \{ (\{A\}, \{B, E\}), (\{A\}, \{C, E\}), (\{B, E\}, \{D\}), (\{C, E\}, \{D\}) \}, \]

and

8. \[ \alpha(W) = (P_W, T_W, F_W). \]
5.

\[ Y_W = \{(\{A\}, \{B, E\}), (\{A\}, \{C, E\}), (\{B, E\}, \{D\}), (\{C, E\}, \{D\})\}, \]

\[ W = \{ABCD, ACBD, AED\} \]

\[ P_W = \{i_W, o_W, p(\{A\}, \{B, E\}), p(\{A\}, \{C, E\}), p(\{B, E\}, \{D\}), p(\{C, E\}, \{D\})\}, \]

\[ F_W = \{(i_W, A), (A, p(\{A\}, \{B, E\})), (p(\{A\}, \{B, E\}), B) \ldots (D, o_W)\} \]

\[ \checkmark 6. \quad P_W = \{p_{(A, B)} \mid (A, B) \in Y_W\} \]

\[ \checkmark 7. \quad F_W = \{(a, p_{(A, B)}) \mid (A, B) \in Y_W \} \]

\[ \cup \quad \{(p_{(A, B)}, b) \mid (A, B) \in Y_W \} \]

\[ \cup \quad \{(i_W, t) \mid t \in T_i \} \cup \{(t, o_W) \mid t \in T_i \} \]

\[ \checkmark 8. \quad \alpha(W) = (P_W, T_W, F_W). \]
Exercise (2)

- What does the alpha algorithm produce for a log consisting only of the following traces?
  - ACD
  - BCE

Let $W$ be a workflow log over $T$. $\alpha(W)$ is defined as follows.

1. $T_W = \{ t \in T \mid \exists \sigma \in W \ t \in \sigma \}$,
2. $T_I = \{ t \in T \mid \exists \sigma \in W \ t = \text{first}(\sigma) \}$,
3. $T_O = \{ t \in T \mid \exists \sigma \in W \ t = \text{last}(\sigma) \}$,
4. $X_W = \{ (A,B) \mid A \subseteq T_W \land A \neq \emptyset \land B \subseteq T_W \land B \neq \emptyset \land \forall a \in A \forall b \in B \ a \rightarrow_W b \land \forall a_1,a_2 \in A \ a_1 \#_W a_2 \land \forall b_1,b_2 \in B \ b_1 \#_W b_2 \}$,
5. $Y_W = \{ (A,B) \in X \mid \forall (A',B') \in X \ A \subseteq A' \land B \subseteq B' \Rightarrow (A,B) = (A',B') \}$,
6. $P_W = \{ p_{(A,B)} \mid (A,B) \in Y_W \} \cup \{ i_W,o_W \}$,
7. $F_W = \{ (a,p_{(A,B)}) \mid (A,B) \in Y_W \land a \in A \} \cup \{ (p_{(A,B)},b) \mid (A,B) \in Y_W \land b \in B \} \cup \{ (i_W,t) \mid t \in T_I \} \cup \{ (t,o_W) \mid t \in T_O \}$, and
8. $\alpha(W) = (P_W,T_W,F_W)$. 

• Direct succession: $x>y$ iff for some case $x$ is directly followed by $y$.
• Causality: $x\rightarrow y$ iff $x>y$ and not $y>x$.
• Parallel: $x\parallel y$ iff $x>y$ and $y>x$.
• Choice: $x\# y$ iff not $x>y$ and not $y>x$. 

•
Exercise (3)

• What does the alpha algorithm produce for a log consisting only of the following traces?
  • ACEG
  • AECG
  • BDFG
  • BFDG

Let $W$ be a workflow log over $T$. $\alpha(W)$ is defined as follows.
1. $T_W = \{ t \in T \mid \exists \sigma \in W \ t \in \sigma \}$,
2. $T_I = \{ t \in T \mid \exists \sigma \in W \ t = \text{first} (\sigma) \}$,
3. $T_O = \{ t \in T \mid \exists \sigma \in W \ t = \text{last} (\sigma) \}$,
4. $X_W = \{ (A,B) \mid A \subseteq T_W \land A \neq \emptyset \land B \subseteq T_W \land B \neq \emptyset \land \forall a \in A \forall b \in B \ a \rightarrow_W b \land \forall a_1, a_2 \in A \ a_1 \#_W a_2 \land \forall b_1, b_2 \in B \ b_1 \#_W b_2 \}$,
5. $Y_W = \{ (A,B) \in X \mid \forall (A',B') \in X A \subseteq A' \land B \subseteq B' \Rightarrow (A,B) = (A',B') \}$,
6. $P_W = \{ p_{(A,B)} \mid (A,B) \in Y_W \} \cup \{ i_W, o_W \}$,
7. $F_W = \{ (a,p_{(A,B)}) \mid (A,B) \in Y_W \land a \in A \} \cup \{ (p_{(A,B)}, b) \mid (A,B) \in Y_W \land b \in B \} \cup \{ (i_W, t) \mid t \in T_I \} \cup \{ (t, o_W) \mid t \in T_O \}$, and
8. $\alpha(W) = (P_W, T_W, F_W)$. 

• Direct succession: $x>y$ iff for some case $x$ is directly followed by $y$.
• Causality: $x\rightarrow y$ iff $x>y$ and not $y>x$.
• Parallel: $x||y$ iff $x>y$ and $y>x$.
• Choice: $x\# y$ iff not $x>y$ and not $y>x$. 

Properties of the Alpha algorithm

- If log is complete with respect to relation $>$, it can be used to mine any SWF-net!
- *Structured Workflow Nets* (SWF-nets) have no implicit places and the following two constructs cannot be used:

(Short loops require some refinement but not a problem.)
Alpha algorithm

• Mainly of theoretical interest!
• Too simple to be applicable to real-life logs.
• Does not address issues such as noise, etc.
• Should NOT be taken as a benchmark.
• However, the algorithm reveals:
  • basic process mining ideas and concepts in 8 lines,
  • theoretical limits of process mining.
Basic test for any mining algorithm: Rediscovery

Can the mined process generate all the behavior in the log?

How close is the behavior of the mined process to the original one?
Controlled choices cannot be rediscovered (and in many cases this is good!)

Fig. 7. The nonfree-choice WF-net $N_6$ cannot be rediscovered by the $\alpha$ algorithm.
Log only contains information about behavior and not structure

Fig. 8. WF-net $N_8$ cannot be rediscovered by the $\alpha$ algorithm. Nevertheless, $\alpha$ returns a WF-net which is behavioral equivalent.
Completeness notion may be too crude in some cases

Fig. 9. Although both WF-nets are not behavioral equivalent they are identical with respect to $>$. 
Another example of behaviorally equivalent SWF-nets

Fig. 10. Both SWF-nets are behavioral equivalent and, therefore, any algorithm will be unable to distinguish $N_{12}$ from $N_{13}$ (assuming a notion of completeness based on $>$).
Silent steps (and duplicate steps) cannot be discovered
(b) Process model

**fitness** +
**precision** +
**generalization** +
**structure** +

**fitness** +
**precision** -
**generalization** +
**structure** +

---

(a) Event Log

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>1207</td>
<td>ABDEI</td>
</tr>
<tr>
<td>145</td>
<td>ACDGHFI</td>
</tr>
<tr>
<td>56</td>
<td>ACGDHFI</td>
</tr>
<tr>
<td>23</td>
<td>ACHDFI</td>
</tr>
<tr>
<td>28</td>
<td>ACDHF1</td>
</tr>
</tbody>
</table>

---

(c) Process model

---

(d) Process model

---

(e) Process model

---

**fitness** -
**precision** +
**generalization** -
**structure** +

**fitness** +
**precision** +
**generalization** -
**structure** -
Simple process mining algorithms tend to:

- Have problems with complex control-flow constructs. For example, many process mining algorithms are unable to deal with non-free-choice constructs and complex nested loops.
- Not allow for duplicates. In the event log it is not possible to distinguish between activities that are logged in a similar way, i.e., there are multiple activities that have the same “footprint” in the log. As a result, most algorithms map these different activities onto a simple activity thus making the model incorrect or counter-intuitive.
- Silent steps. Things that are not recorded cannot be discovered.
- Underfit (i.e., overgeneralize) or overfit. Many algorithms have a tendency to overgeneralize, i.e., the discovered model allows for much more behavior than actually recorded in the log. In some circumstances this may be desirable. However, there seems to be a need to flexibly balance between “overfitting” and “underfitting”.
- Yield inconsistent models. For more complicated processes many algorithms have a tendency to produce models that may have deadlocks and/or livelocks. It seems vital that the generated models satisfy some soundness requirements (e.g., the soundness property).
Other Process Discovery Techniques
Overview of process discovery techniques

• Classical techniques (e.g., learning state machines and the theory of regions): cannot handle concurrency and/or do not generalize (i.e., if it did not happen, it cannot happen).

• Algorithmic techniques
  • Alpha miner
  • Alpha+, Alpha++, Alpha#
  • Heuristic miner
  • Multi phase miner
  • ...

• Genetic process mining

• Region-based process mining
  • State-based regions
  • Language based regions
Genetic Mining
(Ana Karla Alves de Medeiros et al.)

1. initial population
2. fitness test
3. select best parents
4. crossover
5. children
6. mutation
7. new population
Design choices

**fitness**

1. initial population
2. fitness test
3. select best parents
4. crossover
5. children
6. mutation
7. new population

**representation**

**crossover**
Properties of Genetic Mining

• Requires a lot of computing power.
• Can deal with noise, infrequent behavior, duplicate tasks, invisible tasks, etc.
• Allows for incremental improvement and combinations with other approaches (heuristics post-optimization, etc.).
Balancing Between Overfitting and Underfitting
Challenge: Balancing Between Underfitting and Overfitting
The essence

ABCD
ACBD
AED
ABCD
ABCD
AED
ACBD
...

A

E

B

C

D
Any log containing activities A, B, C, D, and E.
Finding a balance

(a)

ACD
BCE
...

(b)

ACD
ACE
BCE
BCD
...

(c)

more
behavior

(d)

more
behavior
ACD 99
ACE 0
BCE 85
BCD 0
ACD 99
ACE 88
BCE 85
BCD 78
Important observations

• Frequencies matter!
• Adding a place equals restricting behavior!
• "The model" does not exist!
Relevance

See examples in Part I  !!!!!
Discovering Other Perspectives
Perspectives

- Control-flow perspective
  - As before ...

- Data perspective
  - How does data flow from one task to another?
  - What data is influencing decisions?
  - What are the (data-driven) business rules?

- Organizational perspective
  - Who is doing what?
  - Who is working with who?
  - What are the (real) roles in an organization?

- ...

...
Examples of social network mining
(Minseok Song et al.)

(b) working together
Social networks based on hand-over of work

(a) social network for organizational units

(b) social network for roles
Decision mining
(Anne Rozinat et al.)
Conformance and Extension
Conformance checker
(Anne Rozinat et al.)

How to quantify this?
Fitness by replay

\[ m = \text{missing}, r = \text{remaining}, c = \text{consumed}, p = \text{produced} \]
No problem (m=0, r=0)

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>1207</td>
<td>ABD(\text{E})A</td>
</tr>
<tr>
<td>145</td>
<td>ACDGHFA</td>
</tr>
<tr>
<td>56</td>
<td>ACGDHFA</td>
</tr>
<tr>
<td>23</td>
<td>ACHD(\text{E})A</td>
</tr>
<tr>
<td>28</td>
<td>ACDH(\text{E})A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>1207</td>
<td>ABD(\text{E})A</td>
</tr>
<tr>
<td>145</td>
<td>ACDGHFA</td>
</tr>
<tr>
<td>56</td>
<td>ACGDHFA</td>
</tr>
<tr>
<td>23</td>
<td>ACHD(\text{E})A</td>
</tr>
<tr>
<td>28</td>
<td>ACDH(\text{E})A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>1207</td>
<td>ABD(\text{E})A</td>
</tr>
<tr>
<td>145</td>
<td>ACDGHFA</td>
</tr>
<tr>
<td>56</td>
<td>ACGDHFA</td>
</tr>
<tr>
<td>23</td>
<td>ACHD(\text{E})A</td>
</tr>
<tr>
<td>28</td>
<td>ACDH(\text{E})A</td>
</tr>
</tbody>
</table>
Another (impossible) trace

(a)

(b) m = 0
r = 0
c = 0
p = 1

(c) m = 1
r = 0
c = 2
p = 4

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>1207</td>
<td>ABDEA</td>
</tr>
<tr>
<td>145</td>
<td>ACDGHFA</td>
</tr>
<tr>
<td>56</td>
<td>ACGDHFA</td>
</tr>
<tr>
<td>23</td>
<td>ACHDFA</td>
</tr>
<tr>
<td>28</td>
<td>ACDHFA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>1207</td>
<td>ABDEA</td>
</tr>
<tr>
<td>145</td>
<td>ACDGHFA</td>
</tr>
<tr>
<td>56</td>
<td>ACGDHFA</td>
</tr>
<tr>
<td>23</td>
<td>ACHDFA</td>
</tr>
<tr>
<td>28</td>
<td>ACDHFA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>1207</td>
<td>ABDEA</td>
</tr>
<tr>
<td>145</td>
<td>ACDGHFA</td>
</tr>
<tr>
<td>56</td>
<td>ACGDHFA</td>
</tr>
<tr>
<td>23</td>
<td>AcHDF</td>
</tr>
<tr>
<td>28</td>
<td>ACDHFA</td>
</tr>
</tbody>
</table>
Fitness calculation

\[ f = \frac{1}{2} (1 - \sum_{i=1}^{k} \frac{n_i m_i}{\sum_{i=1}^{k} n_i c_i}) + \frac{1}{2} \left(1 - \sum_{i=1}^{k} \frac{n_i r_i}{\sum_{i=1}^{k} n_i p_i}\right) \]

\[ f(M1, L2) = \frac{1}{2} \left(1 - \frac{51}{10666}\right) + \frac{1}{2} \left(1 - \frac{51}{10666}\right) \approx 0.995 \]
Examples

(a) Event Log L1

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>4070</td>
<td>ABDEA</td>
</tr>
<tr>
<td>245</td>
<td>ACDGHFA</td>
</tr>
<tr>
<td>56</td>
<td>ACGDHFA</td>
</tr>
</tbody>
</table>

(b) Event Log L2

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>1207</td>
<td>ABDEA</td>
</tr>
<tr>
<td>145</td>
<td>ACDGHFA</td>
</tr>
<tr>
<td>56</td>
<td>ACGDHFA</td>
</tr>
<tr>
<td>23</td>
<td>ACHDFH</td>
</tr>
<tr>
<td>28</td>
<td>ACDHFA</td>
</tr>
</tbody>
</table>

(c) Event Log L3

<table>
<thead>
<tr>
<th>No. of Instances</th>
<th>Log Traces</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>BDE</td>
</tr>
<tr>
<td>7</td>
<td>AABHF</td>
</tr>
<tr>
<td>15</td>
<td>CHF</td>
</tr>
<tr>
<td>6</td>
<td>ADBE</td>
</tr>
<tr>
<td>1</td>
<td>ACGDFAA</td>
</tr>
<tr>
<td>8</td>
<td>ABEDA</td>
</tr>
</tbody>
</table>

\[ f = 1.000 \quad \text{for L1} \quad f = 0.995 \quad \text{for L2} \quad f = 0.540 \quad \text{for L3} \]
Diagnostics

Process Model M1 after replay of Event Log L2
Other Metrics

• Fitness is not sufficient: hence other metrics are needed such as behavioral and structural appropriateness, etc.

• These metrics cover aspects such as:
  • Punishing for "too much" behavior.
  • Punishing for "overly complex" models.
Extension

- Existing models can be enriched by logs analysis (e.g., indicating bottlenecks, etc.).
- Process mining results can be combined.
- Can be used to create comprehensive simulation models and export them to e.g. CPN Tools:
Example: Log from Dutch municipality

Results of automatically generated CPN Tools simulation models

(a) execution time

(b) waiting time
Part III
Hands-on with ProM

www.processmining.org
Download/install ProM 5.2

www.processmining.org
Exercise (4)

- Consider the following log:
  - a b c d f
  - a c b d f
  - a b d c f
  - a b d c f
  - a c d b f
  - a c d b f
  - a d e f
  - a d e f
  - a e d f

- Use the Alpha algorithm/ProM to discover the corresponding process model (Filename: exercise4.mxml)
Exercise (5)

- Consider the following log:
  - a b c d e f b d c e g
  - a b d c e g
  - a b c d e f b c d e f b d c e g

- Use the Alpha algorithm/ProM to discover the corresponding process model (Filename: exercise5.mxmi)
Exercise (6)

• Consider the following log:
  • a b e f
  • a b e c d b f
  • a b c e d b f
  • a b c d e b f
  • a e b c d b f

• Use the Alpha algorithm/ProM to discover the corresponding process model (Filename: exercise6.mxml)
Exercise (7)

- Load exercise7.mxml
- Inspect the log (e.g. using the internet explorer)
- Use the Alpha algorithm to discover the corresponding process model.
Explore ProM using exercise7.mxml

- **Heuristics miner**
  - discover process model and play with settings

- **Alpha algorithm plugin**
  - discover process model and play with settings

- **Fuzzy Miner**
  - discover process model and explore the various views
  - animate the model

- **Social Network Miner**
  - discover the social network
Explore ProM using exercise7.mxml

- [Woflan Analysis] verify correctness of model
- [Conformance Checker] check the conformance of the mined model
- [Conformance Checker] modify the log (delete and insert events) and the check the conformance
- [Performance Analysis with Petri net] analyze the performance (where are the bottlenecks)
- [Basic Performance Analysis] analyze the performance (explore all options)
- [(Advanced) Dotted Chart Analysis] construct dotted charts and explore all options
Explore ProM using exercise7.mxml (3)

- Convert discovered Petri net into EPC model
- Convert discovered Petri net into YAWL model
- Convert discovered Petri net into heuristic net
If time is left, ....

• Repeat the process using exercise8.mxml
• Repeat the process using repairexample.zip and repairexamplesample2.zip
• Use the above two files to follow the steps described in the ProM Framework Tutorial
Conclusion
Conclusion

• The abundance of event data enables a wide variety of process mining techniques ranging from process discovery to conformance checking.

• A reality check for people that are involved in process modeling.

• Great application possibilities!

• Good research/PhD topic!

• TomTom functionality is already possible today!

• Check out ProM with its 250+ plug-ins
Thanks! cf. www.processmining.org

- Wil van der Aalst
- Peter van den Brand
- Boudewijn van Dongen
- Christian Günther
- Eric Verbeek
- Ana Karla Alves de Medeiros
- Anne Rozinat
- Minseok Song
- Ton Weijters
- Remco Dijkman
- Gianluigi Greco
- Antonella Guzzo
- Kristian Bisgaard Lassen
- Ronny Mans
- Jan Mendling
- Vladimir Rubin
- Nikola Trcka
- Irene Vanderfeesten
- Barbara Weber
- Lijie Wen
- Mercy Amiyo
- Carmen Bratosin
- Toon Calders
- Jorge Cardoso
- Ronald Crooy
- Florian Gottschalk
- Monique Jansen-Vullers
- Peter Khisa Wakholi
- Nicolas Knaak
- Sven Lambrechts
- Joyce Nakatumba
- Mariska Netjes
- Mykola Pechenizkiy
- Maja Pesic
- Hajo Reijers
- Stefanie Rinderle
- Domenico Saccà
- Helen Schonenberg
- Marc Voorhoeve
- Jianmin Wang
- Jan Martijn van der Werf
- Martin van Wingerden
- Jianhong Ye
- Huub de Beer
- Elena Casares
- Alina Chipaila
- Walid Gaaloul
- Martijn van Giessel
- Shaifali Gupta
- Thomas Hoffmann
- Peter Hornix
- René Kerstjens
- Ralf Kramer
- Wouter Kunst
- Laura Maruster
- Andriy Nikolov
- Adarsh Ramesh
- Jo Theunissen
- Kenny van Uden
- ...
Relevant WWW sites

- http://www.processmining.org
- http://www.workflowpatterns.com
- http://www.workflowcourse.com
- http://www.vdaalst.com

http://www.senternovem.nl/innovatieveouchers
MKB 2.500 – 7.500 euro