Discovering Petri Nets

It’s a kind of magic ...

Prof.dr.ir. Wil van der Aalst
Eindhoven University of Technology
Department of Information and Technology
P.O. Box 513, 5600 MB Eindhoven
The Netherlands
w.m.p.v.d.aalst@tm.tue.nl
Outline

• Context: Closing the BPM loop
• Overview of process mining
• ProM toolset
• The alpha algorithm
• Alternative approaches
• Conclusion
The BPM life-cycle

- Diagnosis
- Process design
- Implementation / configuration
- Process enactment

2008
2003
1998
1993

Press arrow to start

2003 © Wil van der Aalst, Vincent Almering
Process mining: Reversing the process

- Process mining can be used for:
  - Process discovery (What is the process?)
  - Delta analysis (Are we doing what was specified?)
  - Performance analysis (How can we improve?)

www.processmining.org
Process mining (overview)

1) basic performance metrics

2) process model

3) organizational model

4) social network

5) performance characteristics

6) auditing/security

If … then …
workflow management systems
- Staffware
- InConcert
- MQ Series

case handling / CRM systems
- FLOWer
- Vectus
- Siebel

ERP systems
- SAP R/3
- BaaN
- Peoplesoft

common XML format for storing/exchanging workflow logs

Core
- input/output
- visualization
- analysis

Plugins
- alpha algorithm
- genetic algorithm
- Tsinghua alpha algorithm
- Multi phase algorithms
- social network miner
- case data extraction
- property verifier

External Tools
- NetMiner
- Viscovery
Let us focus on mining process models …

1) basic performance metrics

2) process model

3) organizational model

4) social network

5) performance characteristics

6) auditing/security

If … then …
Process Mining
Process log

- Minimal information in log: case id’s and task id’s.

- Additional information: event type, time, resources, and data.

- In this log there are three possible sequences:
  - ABCD
  - ACBD
  - EF

<table>
<thead>
<tr>
<th>Case</th>
<th>Task</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 3</td>
<td>task B</td>
</tr>
<tr>
<td>Case 1</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 E</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 F</td>
</tr>
<tr>
<td>Case 4</td>
<td>task D</td>
</tr>
</tbody>
</table>
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

A > B
A > C
B > C
B > D
C > B
C > D
E > F

| B || C | C || B | A → B | A → C | B → D | C → D | E → F |
Basic idea (1)

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

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

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

\[ x \rightarrow z, \ y \rightarrow z, \text{ and } x \parallel y \]
Basic idea (5)

\[ x \rightarrow z, \quad y \rightarrow z, \quad \text{and} \quad x \# y \]
Let $W$ be a workflow log over $T$. $\alpha(W)$ is defined as follows.

$$T_W = \{ t \in T \mid \exists \sigma \in W \ t \in \sigma \} ,$$

$$T_I = \{ t \in T \mid \exists \sigma \in W \ t = \text{first}(\sigma) \} ,$$

$$T_O = \{ t \in T \mid \exists \sigma \in W \ t = \text{last}(\sigma) \} ,$$

$$X_W = \{ (A,B) \mid A \subseteq T_W \land B \subseteq T_W \land \forall a \in A \forall b \in B \ a \rightarrow_W b \land \forall a_1,a_2 \in A \land b_1,b_2 \in B \ a_1 \neq_W a_2 \land \forall b_1,b_2 \in B \ b_1 \neq_W b_2 \} ,$$

$$Y_W = \{ (A,B) \in X \mid \forall (A',B') \in X \ A \subseteq A' \land B \subseteq B' \Rightarrow (A,B) = (A',B') \} ,$$

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

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

and

$$\alpha(W) = (P_W,T_W,F_W).$$

The alpha algorithm has been proven to be correct for a large class of free-choice nets.
Challenges

• Refining existing algorithm for (control-flow/process perspective)
  – Hidden tasks
  – Duplicate tasks
  – Non-free-choice constructs
  – Loops
  – Detecting concurrency (implicit or explicit)
  – Mining and exploiting time
  – Dealing with noise
  – Dealing with incompleteness

• Mining other perspectives (data, resources, roles, …)
• Gathering data from heterogeneous sources
• Visualization of results
• Delta analysis
Alternative approaches

• Something based on the Theory of Regions? (Note that we typically see only a fragment of all possible behaviors of the system, cf. Occam’s Razor.)

• Current approaches:
  – The Tsinghua variant
  – Heuristics (cf. Thumb)
  – Multi-phase process mining
  – Genetic algorithms
Alternative: Multi-phase process mining (Boudewijn van Dongen)

Two phases:

1) Create a visual description of each instance, without choices and loops (cf. runs or occurrence nets).
   - Comprehensive representation
   - Ideal for performance analysis (cf. ARIS PPM)

2) Aggregate multiple instances to one process model.
   - Only causal relations between tasks are required

Properties:

- More robust and multi-lingual (cf. EPCs).
- Possibility of inspect instances
Example

Log file:

A, B, C, B, C, D, E, F, G
A, B, E, D, F, G
A, B, D, E, F, H

Causal relations:

For each entry in every instance, find the closest causal predecessor and successor, and build instance graphs
Translation

Instance graph:

```
A -> B -> C -> B -> C -> D -> F -> G
```

Transformation to EPC:
Three instance graphs:

A → B → C → D → F → G
A → B → D → F → G
A → B → D → E

Aggregated instances:

A → B → C → D → F → G

Transformation to EPC:
Alternative: Genetic Mining
(Ana Karla Alves de Medeiros and Ton Weijters)

• Use genetic algorithms to:
  – mine process models from incomplete event logs
  – Tackle structural constructs (duplicate tasks, non-free choice etc.) that $\alpha$-algorithm does not

• Issues:
  – internal representation of individuals
  – fitness measure
  – genetic operations (crossover and mutation)
Overview of approach

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Read event log</td>
</tr>
<tr>
<td>II</td>
<td>Calculate dependency relations among workflow model elements</td>
</tr>
<tr>
<td>III</td>
<td>Build the initial population</td>
</tr>
<tr>
<td>IV</td>
<td>Calculate individuals' fitness</td>
</tr>
<tr>
<td>V</td>
<td>Stop and return the fittest individuals?</td>
</tr>
<tr>
<td>VI</td>
<td>Create next population - use genetic operations</td>
</tr>
<tr>
<td>V</td>
<td>( \text{yes} )</td>
</tr>
<tr>
<td>V</td>
<td>( \text{no} )</td>
</tr>
<tr>
<td>end</td>
<td></td>
</tr>
</tbody>
</table>

Flowchart:
- Start → I → II → III → IV → V → yes → end
- V → no → VI
- VI → V → yes
- V → no
- Start
Example

Log:
As,Ac,Bs,Bc,Cs,Cc,Ds,Dc
As,Ac,Bs,Bc,Cs,Cc,Ds,Dc
As,Ac,Cs,Cc,Bs,Bc,Ds,Dc
As,Ac,Cs,Cc,Bs,Bc,Ds,Dc

This log is incomplete if we assume that B and C are in parallel but never happen to overlap.
If we only consider the start or complete events, the result is OK. If not, more traces are needed of infer that B and C are in parallel.
Genetic mining result

Fitness 100%
Process mining provides many interesting challenges for scientists, customers, users, managers, consultants, and tool developers.

The alpha algorithm is an example of an approach focusing on the process perspective. It is related to but also quite different from the Theory of Regions. Interesting alternatives to the basic alpha algorithm are being developed.
More information
http://www.workflowcourse.com
http://www.workflowpatterns.com
http://www.processmining.org