A Statistical Variant of the Inductive Miner

Martin Bauer

Institut fr Informatik
Humboldt-Universitt zu Berlin

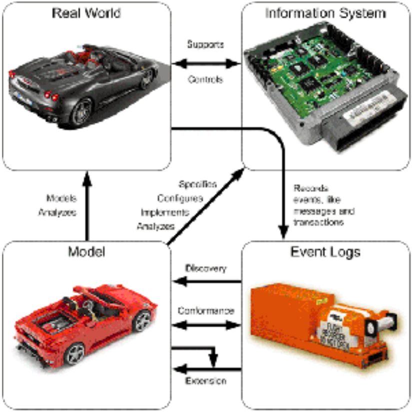
16.6.2017

Contents of this presentation

- Process Mining
- Incremental Log Analysis
- the Statistical Inductive Miner infrequent
- Discussion

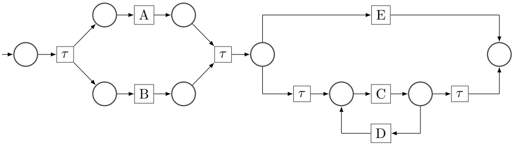
Process Mining

user id	activity	timestamp
3456	A	03-07-2016
3456	B	03-07-2016
4788	A	04-07-2016
3456	C	04-07-2016
4788	D	04-07-2016



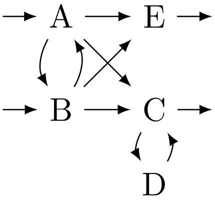
Process Discovery

user id	activity
1	A
1	B
1	E
2	B
2	A
2	C
3	B
3	A
3	C
3	D
3	C

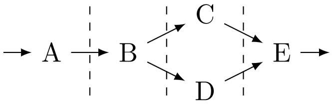


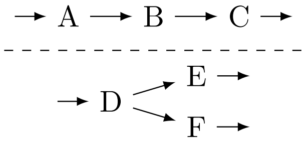
Inductive Miner - Conversion to df-Graph

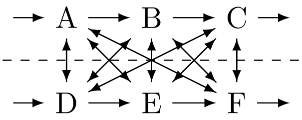
user id	activity
1	A
1	B
1	E
2	B
2	A
2	C
3	B
3	A
3	C
3	D
3	C

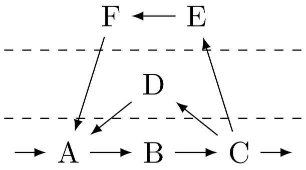


Inductive Miner - Detecting Cuts

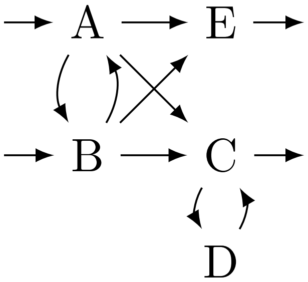


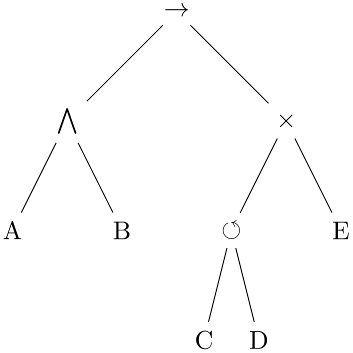




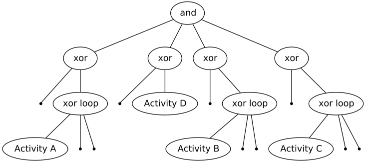


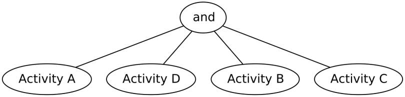
Inductive Miner - Building a Process Tree





Inductive Miner infrequent - Filtering out Noise





Incremental Process Discovery - Redundancy of Logs

- A,B,C,D - 500
- A,B,E - 400
- A,G - 20

How likely are we to see new information in the next N Traces?

$$Bin(N,k) = \binom{N}{k} \cdot p^k \cdot (1-p)^{N-k}$$

Delta-Completeness

"A Log is δ-complete, if the probability p of seeing new information in the next trace is smaller than δ"

Incremental Process Discovery - Estimating p

$$[\frac{1}{1+\frac{z^2}{N}} \cdot (\hat{p} + \frac{z^2}{2N} - \sqrt{\frac{\hat{p} \cdot (1-\hat{p})}{N} + \frac{z}{4N^2}}),$$
$$\frac{1}{1+\frac{z^2}{N}} \cdot (\hat{p} + \frac{z^2}{2N} + \sqrt{\frac{\hat{p} \cdot (1-\hat{p})}{N} + \frac{z}{4N^2}})]$$

Assuming $\hat{p} = 0$ gives

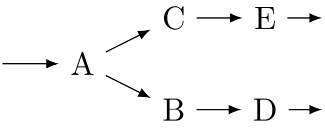
$$\frac{1}{1+\frac{z^2}{N}} \cdot (\frac{z^2}{2 \cdot N} + \sqrt{\frac{z}{4 \cdot N^2}})$$

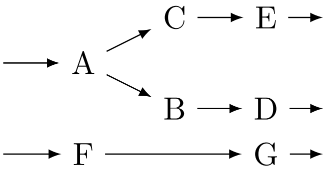
Increase N until upper bound is smaller than δ

Content of the Thesis

- Apply the Incremental Process Discovery Procedure to Inductive Miner infrequent using model correctness and approximated average cycle time as quality criteria
- Evaluate the algorithm on runtime properties and on factors that influence it's effiency

Model Correctness





A trace may add events, edges, starting- or ending nodes - if it does, it contains new information

Cycle Time Correctness

Trace-based and Event-based Cycle times

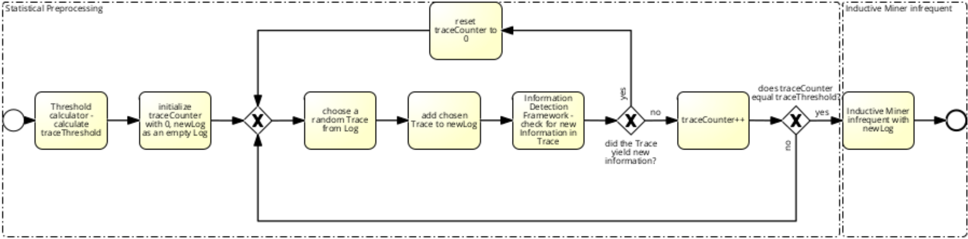
- A(5s), B(10s), C(5s)
- A,B,C (20s)

Iterative Estimation of cycle time is converging towards correct cycle time:

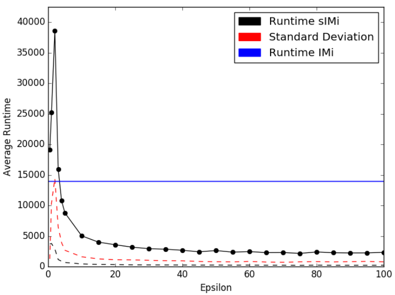
$\forall \epsilon \geq 0 \exists 0 < N < \max(|t|) \forall N \leq t_n, t_m < \max(|t|) : d(cycletime_n, cycletime_m)$

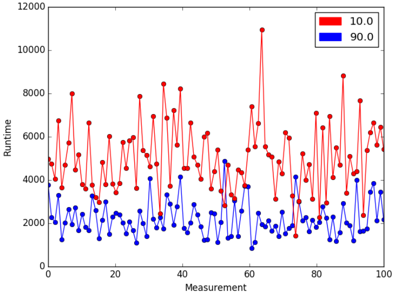
given ϵ, if trace changes cycle time more than ϵ assume new information

The Algorithm

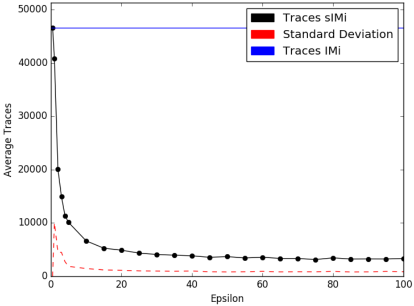


Evaluation - Runtime

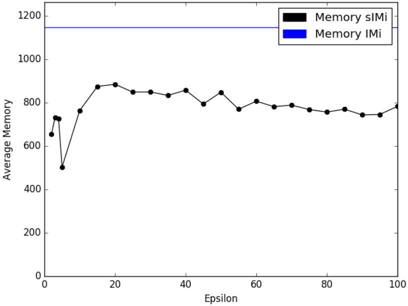




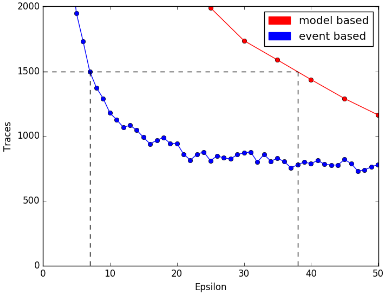
Evaluation - Traces

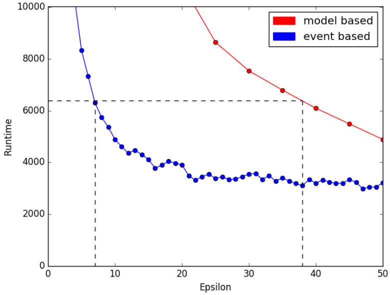


Evaluation - Memory

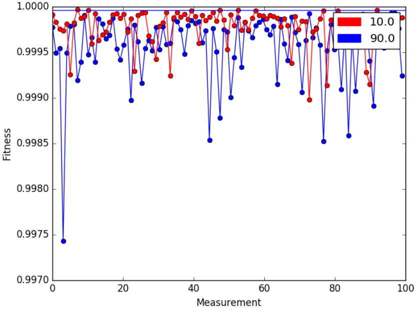


Evaluation - what runtime approximation is better?

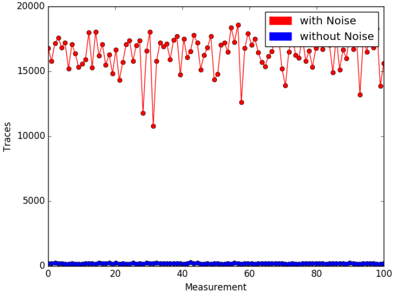


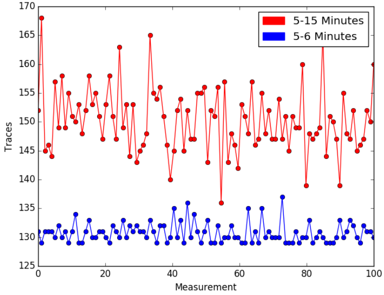


Evaluation - Fitness

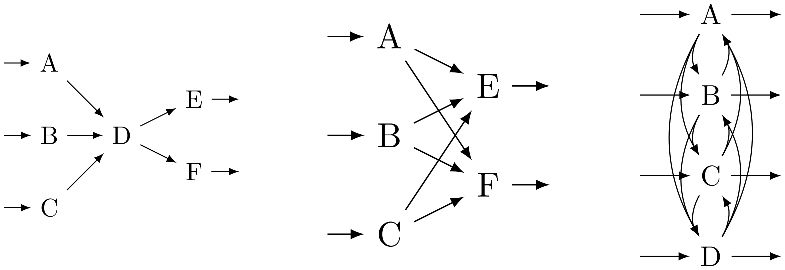


Log properties - Noise and Cycle Time Deviations





Control flow structures



AND and sequential complex blocks (AND, XOR) influence runtime the most

Conclusion

- sIMi outperforms IMi an almost all cases
- the algorithmic framework is extensible
- no fitness loss

- bad ϵ increase runtime by a lot
- random nature of trace picking (can be adjusted easily)

Thank you!
Any questions?