Master Thesis Proposal

Kefang Ding RWTH Aachen Informatik Date: 02. August 2018

Proposed Topic:

Repair Log Based on Real Behavoir

Topic

Real behavior in companies, as recorded during process execution, usually deviates from the ~~existing~~ process model. To improve the performance of ~~process~~, one approach is to repair the existing process model according to observed behavior, which is recorded in event log. However, the current methods for model repair have some shortcomings:

1) the classification of traces is not good enough. Association rules neither decision tree quantifies the goodness of traces on performance improvement.
2) the improvement measure ~~only bases~~ on one KPI at once;
3) only positive traces in ~~event log~~ are used to repair the model;
4) they repair the model by adding loops, subprocess, and removing transitions. However, they can't deal with the reordering of transitions, concurrency change(transition relation from sequential to concurrent); Besides, adding loop is an separate work from adding subprocess, such that it demands extra alignment computation.
5) by adding silent transitions and loops, a less precise model is generated.

In my master thesis, I would like to analyze the current model repair methods and overcome some of their drawbacks to improve model repair.

After the first analysis, I have gotten another idea for a topic. But not sure if it works.

(6)* Process Discovery with Neural Network

My inspiration comes from the graph structure of Petri net and the training of neural network, when I try to use positive as well as negative traces for model repair.

In this model, every node has a chance to connect to others(constrains will be expressed in weights on the node or the arcs). The node has a threshold function to enable different transition in the next step, which triggers a path through this network.

After reading the paper about process discovery of Artificial Negative Events, my idea and theirs have common thought.

Methodology:

Inputs:

- Existing process model in Petri net
- Event Log in real execution of process
- Key Performance Indicators(KPI)

The approach is described in the graph below. Process in solid black line is the framework from Fahland; dashed line are the parts for performance improvement; green parts are the improvement I propose.

The framework works in this way. Firstly, ~~event log~~ and the existing process model are given as input for deviation analysis. After finding the deviations, ~~event log~~ is divided into different clusters for log repair. After that, model repair is conducted for each cluster ~~by adding loops and add subprocess.~~ At end, it removes the infrequent paths from process model. To improve model performance, Dees and Syring have added decision rules to classify the examples into positive and negative; however, only positive examples are passed to the cluster step, while negative examples are left out.

In my suggested parts in green, the negative examples are also used to improve the model in the later step. Besides, I try to quantify the different impact of traces by using regression function on performance improvement and combine more KPI at the same time to get better total improvement. The parts, regression,

multiple KPI measurement and combining negative impact ~~should be pluggable, by this, I mean, it could combine into another methods to repair model~~.

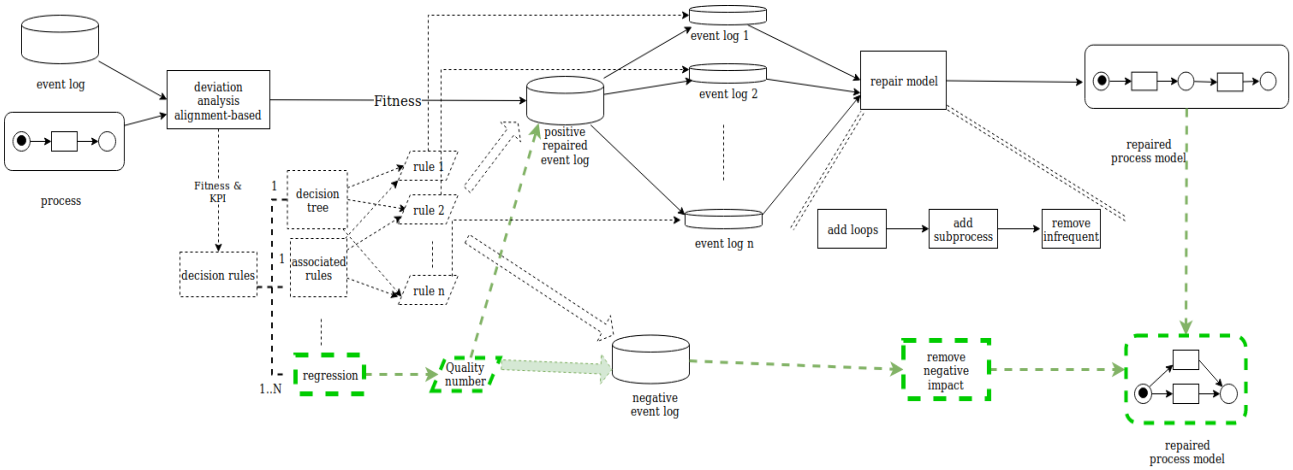

Figure 1: approach to repair model

Next, I will divide my work into the following sub tasks and give more details.

(1) change the classification

To separate the cases, they consider only the information of event(the number of events it happens). Based on this, they create splitting rules for positive or negative traces. I'd like to analyze the degree of their impact by giving a number of goodness from a regression function.

(2) combine multiple KPI for improvement

I would like to combine several KPI into one global quality function, firstly using the weight given by experts(higher weight, higher expectation). Then quantity its KPI, not just good or bad. According to the different quality(number), we give corresponding changes on the model. The quality is higher, so the chance to this change is higher.

(3) use the negative examples

- Naive methods:

To events, count the frequency of events sequence in negative examples and remove them later. To sequence, we count the frequency of such paths, and remove nodes during this way.

- Combine AGNEs methods:

Method "Artificial Negative Events (AGNEs)" at first uses TILDE to learn rules from both positive and negative traces. Later it transforms the classification rules to Petri net. Motivated by this method, I improve model by using negative examples in the following way:

- change Petri net into rules patterns.

  Petri net patterns is bijective in the rules of the language bias. So the reverse transformation is feasible.

- train rules based on the negative examples and get modified rules

  More constraints are added from negative examples for each rules;However, it's based on the existing rules, so the similarity to the existing ones should be guaranteed.

- transform the modified rules into Petri net

The main attention should be ~~paid on~~ the transformation from Petri net to rules in logic programm.

- Novel ones by combining the training from neural network

In the existing model, if transition and place are considered as node, every node has a chance to connect to others(constrains will be expressed in weights on the node or the arcs). The weight on the path means the connectivity. The node has a threshold function to trigger different transition in the next step. By this way, we can use negative and positive examples to adjust the weight, and then build the network. More details are in following part (5).

For model repair, I need to adjust the weight on the path, the connectivity. If connectivity is strong, they have strong causal relation(I'd like to say the direct following relation).

Possible changes:

– add more transitions, subprocess.

If we see some transitions not in this model, then we add it to each layer of this network(from the summary of event log). If there is a direct following relation to this event, then we add a connection from this place to it. (How to make sure it is global??We need to add arcs to the trigger of next event? Still some question to consider, consider it is a path, not isolate events)

– change the order or concurrency

It could be done by changing the connectivity of each node. If two nodes from the same source are both strong, they have high probability to happen concurrently.

(4) improve alignment-based method to include the change

The current alignment method considers the synchronous move, log move and model move, but not the situation about moves modification, which could imply the order change. If the alignment method could consider the modification, the reorder function will be extended.

(5) change the model to neural network for process discovery

The model has original layers with number of transitions in the event log. Given one trace, it goes through the model and build the connection of each transition. After comparing to the predicted label(we see it positive or negative), the weight in each layer is adapted. If trace is positive, the connection is reinforced; if negative, the weight decreases on the path.

[Because some process models are full of transitions, so with too many layer, it's hard to train and lots of data are needed. I wonder if it's better to build an original process model(which is from Alpha Algorithm or other methods) and based on this, we use neural network to train on it for saving time and improving efficiency.]

To interpret the model to process, since it's weight based, we could use a filter to show the strong connection and build the models, or at beginning to ignore the noise when building the model. It could transform to Petri net when considering the strong connection.

To check conformance, fitness is calculated by cost through network like Petri net replay. If there is a connection, then it follows it; if not, we may have some empty node in this model, and it follows this node. For performance in this model, we get it by going through the path in this model. The advantages are maybe, I think, we could use both negative and positive examples to train this model, combine the possibility of paths in this model. Some disadvantages are we need more data to train the model.

My first attention is to deal with the problems in (1),(2),(3), since (1)(2) could be solved easier by borrowing ideas from existing machine learning methods and I find (3) quite interesting. Later, I will try to solve the problems in (4)(5).

References / Bibliography:

1. Fahland, Dirk, and Wil MP van der Aalst. "Model repair—aligning process models to reality." *Information Systems* 47 (2015): 220-243.
2. Dees, Marcus, Massimiliano de Leoni, and Felix Mannhardt. "Enhancing process models to improve business performance: a methodology and case studies." *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, Cham, 2017.
3. Adriansyah, Arya, et al. "Alignment based precision checking." *International Conference on Business Process Management*. Springer, Berlin, Heidelberg, 2012.
4. Anja Syring's Master thesis. "Analysis and Improvement of Process Models with Respect to Key Performance Indicators: A Debt Collection Case Study"
5. Adriansyah, Arya, Boudewijn F. van Dongen, and Wil MP van der Aalst. "Towards robust conformance checking." *International Conference on Business Process Management*. Springer, Berlin, Heidelberg, 2010.
6. vanden Broucke, Seppe KLM, et al. "Determining process model precision and generalization with weighted artificial negative events." *IEEE Transactions on Knowledge & Data Engineering* 1 (2013): 1.
7. Buijs, Joos CAM, et al. "Improving business process models using observed behavior." *International Symposium on Data-Driven Process Discovery and Analysis*. Springer, Berlin, Heidelberg, 2012.