Inductive Reference Modelling Based on Simulated Social Collaboration

Andreas Sonntag[1], Peter Fettke[1], Peter Loos[1]

[1] German Research Center for Artificial Intelligence at Saarland University, Institute for Information Systems, Saarbrücken, Germany
{andreas.sonntag,peter.fettke,peter.loos}@iwi.dfki.de

Abstract. Organizations nowadays possess huge repositories of process models. Inductive reference modelling can save costs and time by reusing process parts of process models belonging to a common domain. The inductive development of a reference model for a large corpus of process models is a difficult problem. Quite a few, primarily heuristic approaches have been proposed to the research community that require an approximate matching between the single processes. With our approach, we introduce a new concept that brings in for the first time an abstract efficiency simulation of the social collaboration around knowledge-based process models. A reference model is assembled featuring at least the topological minimum requirements to be significantly more efficient than the input process models. Our evaluation indicates that the approach is able to generate reference process models that are more efficient than the input process models and at least as a reference model designed by an expert.

Keywords: Social Collaboration, Knowledge Work, Social Network Analysis, Reference model mining

1 Introduction

1.1 Research Motivation

Business process models, as event-driven process chains (PMs), are representatives of processes in an enterprise or in an organization. Current processes may be analysed in order to be improved. Organizations maintain huge repositories of these PMs. Therefore, the PMs have to be analysed as it is necessary for optimizing and managing the repositories. Typically, business analysts have specialized knowledge of the processes and provide expertise in modelling reference process models (RMs). These RMs enable reusability, modularity and avoiding redundancy by the design of new models which saves costs and time. They represent a best practice in a domain and can be extended for individual requirements [1]. The inductive development of universally applicable and reusable RMs (reference modelling in the following) is of great importance in the field of business process management as it seeks to infer generic models or patterns of a domain [2]. A strategy of reference modelling is to

13th International Conference on Wirtschaftsinformatik,
February 12-15, 2017, St. Gallen, Switzerland

Sonntag, A.; Fettke, P.; Loos, P. (2017): Inductive Reference Modelling Based on Simulated Social Collaboration, in Leimeister, J.M.; Brenner, W. (Hrsg.): Proceedings der 13. Internationalen Tagung Wirtschaftsinformatik (WI 2017), St. Gallen, S. 701-715

identify similarities between individual PMs and to derive an abstracted PM [3]. In contrast to deductive approaches, general theories and concepts are not applied for the development of RMs [4]. The utilization of RMs can save time and costs towards single PM variants [17] which makes their execution efficient. The efficiency of a RM hence should be considered when aiming at developing RMs.

1.2 Related Work

Prior respecters of reference modelling are: [25] mine RMs by analyzing desired behavior from event log data. [27] and [26] develop a RM that has a minimal graph-edit distance respectively a minimal cost of change to given process variants. [7] and [30] apply genetic algorithms. [32] apply factor analysis to find statistical commonalities in the process structures. [33] assemble a RM from hierarchical clustered process fragment. [28] seek to identify similar frequent substructures among PMs. [29] derive a RM by gathering similarities of given PMs not only by their structure but by their behavior profile. [31] iteratively create a RM based on the proximity of node pairs.

The named previous automatic reference modelling approaches develop a RM "bottom-up" by comparing the node labels or PM topology. This comparison requires critical assumptions such as a certain model design quality, consistent syntax and language. The understanding of the semantics or similarity of process elements is subject to the process matching problem whose underlying graph matching problem is principally NP-complete [16]. There are heuristic approaches for the problem that achieve an approximate solution which is not generally unique and is afflicted with a loss of information [5].

1.3 Research Problem and Approach

Social collaboration in organizations is a competitive advantage as it is a driver for efficiency, time and money saving potential, product quality and knowledge diffusion [15]. Reference modelling has to be seen under several facets simultaneously in regard to quality and usability [18]. We belief that social collaboration around PMs is an important facet for reference modelling, especially in the context of knowledge-based processes. In the field of inductive reference process model mining, a social perspective for matching the process flow is missing. Prior approaches are primarily based on label matchings and similar graph structure detection in the process models. The influence of the topology of social collaboration between the humans that work around the process (performers) is neglected. Our research question thus is, what is an efficient collaboration topology between performers for a corpus of process models.

Therefore, we want to conceptualize and implement in this work an approach for the inductive development of a RM from a corpus of knowledge-based business process models applying a social perspective for matching the process flow. The resulting RM shall consist of an efficient collaboration topology for the performers working at their associated process functions. Following the concept of [9], the RM is a PM derived from a performer network (PN) that represents at least the minimum

requirements of the PM to be efficiently executed. The PN for the RM development is optimized to fit best to the efficiency of the input PMs. The definition of efficiency can be found in section 2.2. The RM development and interpretation does not need a label matching. The resulting RM consists the minimum process elements for being as efficient as the PMs from it was derived. It can be tailored to concrete organizations and processes. Thereby, risks for the execution potential of a PM can be identified and reduced; needed key performers and their influence can be tracked; the structure of successful co-workership and knowledge/information transfer can be disclosed. The organization becomes more transparent and costs as well as managing effort are reducible.

Our approach relies on social network analysis and evolution strategy [10]. Its empirical evaluation follows [8] and the prototypical implementation is based on design research [14]. The validation of our concept faces three scenarios in which our prototypical implementation develops RMs from three PM corpora of different domains and two languages. The developed RMs are compared to the given respective gold RMs of each scenario. Gold RMs are designed by domain experts.

The paper is structured as follows: After fundamental concepts and terms are introduced, our approach is described followed by the validation of our concept with the experimental design, the evaluation and the discussion of our results. Afterwards, a conclusion brings this work to a close.

2 Fundamentals

2.1 Social Network Analysis

A social network represents the relation between agents. Agents are constituted as nodes and the links between them as edges. The agents are atomic units and they can only communicate and collaborate with agents that are directly connected to them. Social network analysis (SNA) seeks to disclose social interactions, political power and co-workership by abstracting from social relationships [6]. A social network has *n* agents/nodes and *m* edges. People tend to form clusters which are groups of directly connected individuals [20]. Those edges are called local edges. Edges between clusters are called global. The extend of this tendency can be measured by the average clustering coefficient of a network, *CC*. *CC* is the number of the actual- in relation to the possible number of edges between an agent's neighbours, averaged over all agents. The number of an agent's neighbours is called degree. The mean degree of a network *md* is the average number of all agents' neighbours. The number of agents on the shortest path between two gents is called path length. The network density *dens* is defined as $\frac{2m}{n*(n-1)}$. It describes the relation between existing and possible edges in the network which makes it a measure for being sparse. Being sparse is an essential property of many social networks as they tend to form relatively small butt dense clusters with few global edges between them [19]. Among others, this makes such social networks a "small world", in which all individuals, e.g. independent from geographic distances, are separated by only a few edges [20]. Another important

peculiarity of social networks is the appearance of hubs as [22] suggested in theory- and [21] in an empirical study as they confirmed the appearance of hubs in a real organization over time. Hubs are nodes that have a much higher degree than the most other nodes. Hubs constitute hierarchy which is a necessary empirical implication of social networks. The outcome of actual relationships strongly depends on structural network properties [23]. The identification of critical joints in the network such as the collaboration of working individuals that are geographically separated and that may belong to different functional- and hierarchical positions, is of strategic importance for collaboration [24].

2.2 Efficiency of Process Models

The concept of performer networks (PNs) and their efficiency is introduced by [9]. Almost every business process needs social collaboration to be efficiently executed [12]. The concept of PNs connects the aim of business process management to design, model and execute efficient processes with the potential of SNA. Performers are agents working on process functions with a set of capabilities in a PN. Capabilities for this study are simplified as a mapping between a process function and a number indicating the extent of being capable/efficient to work at this function.

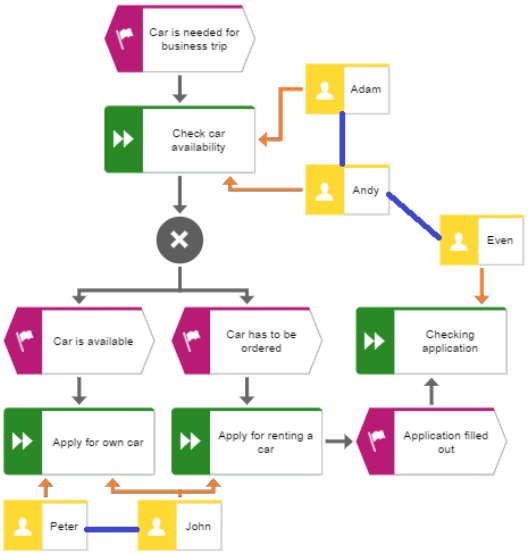

Figure 1. An example PN fragment around a PM

PNs represent a minimal topology of co-workership around business processes in an organization. PNs are formalized as social networks in which exist two kinds of edges: social edges that connect performers and functional edges that connect performers with process functions (*assigned_nodes*). An example PN fragment around a PM is presented in figure 1. The blue edges between the performers (named nodes) are social edges and the black edges/arrows between performers and process

functions are functional edges. The other edges are process edges, connecting process nodes. The efficiency of a PM is the efficiency of the most efficient combination of a generated PN and the PM. The efficiency of a PN/PM combination is computed with an algorithm based on social network analysis and evolution strategy. Several repeated simulations of a work package, processed through the control flow of the PM, count how many iterations were needed to complete the whole process. Only performers associated to a process function (*involved_performers*) process the work package. They are much faster if they have the proper capability for this function. At the same time, neighbours of the performer collaborate which increases the PN efficiency *PNE*. *PNE* corresponds to the standardized PN efficiency definition of [9]. $PNE = 1 - (\frac{mean(\#neededIterations)}{max\#Iterations})$. *PNE* is the inverse of the relation between the number of needed iterations, averaged over all simulation runs $mean(\#neededIterations)$ and the maximal possible number of iterations. $max\#Iterations$ refers to a PN with one incompetent performer assigned to each process function. A PN can be generated by every social network generator that assigns at least one performer to a process function. The evaluation of PNs in [9] implies that a performer topology with hierarchy, short paths and significant clustering generates efficient PNs. Their suggested network generator by [11], PN generation algorithm and parameters are the basis for our PN generation in *generatePN* (see algorithm 5).

3 Approach

For our approach, we consider real-world PMs as event-driven process chains which are directed graph structures, consisting of a set of edges that indicates an order between a set of nodes. Each node has a label, as a linguistic expression of natural language, describing the node's function. Other (meta) information that might be provided is ignored. The fundamental assumption of our approach is that a RM must be efficient at least significantly more efficient than the most efficient input PM. We want to develop a reference PN from the input PMs entailing at least the topological minimum requirements to be efficient around all input PMs. It can be interpreted as a reference for positioning and putting people on team tasks for the whole organization in which all input PMs are assumed to be executed. Our approach's advantage is that no label matching or PM topology matching has to be considered as prior approaches require.

In this section we describe our approach on a high level. The procedure for the development of a RM from a set of input PMs is presented in algorithm 1. It is based on an evolutionary strategy which is an optimization driven only by mutating and selecting individuals [10]. The mutation rate can be increased every run, if no convergence was reached, which allows bigger steps through the solution space. The evolutionary strategy itself runs in linear time and can achieve a fitness/solution quality convergence to an acceptable solution after only a few iterations. The apparent advantage of evolutionary strategy, in contrast to other evolutionary algorithms, is to be fast and parallelizable because the population has only to be evaluated once per

iteration. If it becomes foreseeable after many iterations that the individual's fitness will not improve, converges or even impairs, the evolutionary strategy can be run again or parallel runs on many processors at the same time can be executed until an acceptable solution occurs. The disadvantage on the other hand is that an evolutionary strategy can converge towards a local optimum in only few iterations but there is no proof that a global optimum will ever be reached [13].

Every PN is considered as an individual/possible solution. Its efficiency can only be seen in connection to a PM / or a set of PMs. The fittest resulting individual is the reference PN for the input PMs. In order to ensure reliability, the evolution strategy is repeated 1000 times as this meets the limitation of our computing time. We circumvent the label matching problem by mutating the individuals. The mutation is a random decision about how to arrange information inside an individual. This random decision made during the evolution compensates that we have no distinct knowledge about the distribution of the decision's quality. The selection forces the mutation quality to become better over many iterations. Therefore, a fitness function is defined in algorithm 4 that evaluates the sum of the efficiency of an individual towards all input PMs weighted by the inverse of its density. The efficiency of a PN/PM combination is evaluated with the procedure by [9]. The additional weight is necessary because otherwise a super PN with thousands of performers and functional edges would receive the highest fitness. A small/sparse PN with similar efficiency is to prefer. The same goes for a RM because the more process nodes a RM comprises, the bigger it becomes since there is no matching which would condense multiple nodes. As an objective evaluation criterion, a smaller RM with similar efficiency is likewise to prefer. "Super-models" with many nodes overloaded a user/modeller and prevented him from brining in own ideas [7].

The development of a RM follows the evolutionary strategy in *developReferenceModel* (algorithm 1): At first, for each PM in the set of input PMs *Models* an efficient PN is generated with *generatePN* in algorithm 5. Those PNs represent the initial population and are only optimized to be efficient for their respective PM. The best individual out of this population is picked by the selection operator who calculates all individuals' fitness values and chooses the individual with the highest fitness value. This individual is now called *best_individual* and its fitness is *best_fitness*. In the same step, *cur_fitness* and *minimum_fitness* are set to the same fitness value. The current individual *cur_individual* on which the mutation and selection operators are applied on is now created and initially set to a copy of *best_individual*. *cur_fitness* is the fitness of the current individual. The minimum fitness *minimum_fitness* is the fitness that must be significantly exceeded by *cur_fitness* during the evolution strategy (step 5) in which the mutation and selection operators drive *cur_individual* to reach a fitness convergence significantly greater than the minimum fitness. The mutation of an individual (algorithm 2) assigns random capabilities out of the possible capabilities in the input PMs to a randomly chosen portion of performers (between 0 and 100%) and flips their edges. When the evolution strategy found a good PN, a RM is derived from it (see *deriveEfficientModel* in algorithm 3): The RM contains all process functions to which the performers of the PN are assigned. A process edge between two functions is added

if any pair of performers is connected so that the first performer is assigned to the first function and the other one to the second function. Conversely, that means that a generated PN around the RM is efficient in all combinations with the input PMs.

The developed RM is to understand as a minimum model that exhibits at least the efficiency of the input PMs and simultaneously it has relatively few nodes. By extending the fitness function, the resulting RM can be adapted to concrete requirements. The final step in the RM development would be to specify the performers in the real organization and bringing the reference PN and the developed RM together. The developed RM has to be seen as a combination of a reference PN with its corresponding PM. Only this combination can be interpreted and compared. This is a comparable constraint to a RM developed with the help of a label matching as that RM can only be used if the matching is of suitable quality which is subjective and difficult to measure.

Algorithm 1: Development of a RM:
developReferenceModel(Models)
Input: *Models*, A set of process models Output: A RM

1. Let *best_individual* be a PN = *pn* that maximizes *fitness(pn, Models)* with *pn = generatePN(m)* for each PM *m* ∈ *Models*
2. *best_fitness = cur_fitness = minimum_fitness = fitness(best_individual, Models)*
3. *cur_individual = best_individual*
4. **Repeat** until convergence of *cur _fitness >> minimum_fitness*:
    a. *mutate(cur_individual, mutation_rate, Models)*
    b. *cur_fitness = fitness(cur_individual, Models)*
    c. **If** *cur_fitness > best_fitness* **Then**:
        i. *best_individual = cur_individual*
        ii. *best_fitness = fitness(best_individual, Models)*
5. **return** *deriveEfficientModel(best_individual)*

Algorithm 2: Mutation of a PN: *mutate(PN, mutation_rate, Models)*
Input: A PN, *PN*, *Models* and *mutation_rate* in [0;1]

1. *portion* = a set with (*mutation_rate* * 100%) of *PN.performers*
2. **For Each** *performer* in *portion* **Do**:
    a. *performer.capabilities* = random capabilities from *Models*
3. **For Each** edge *(p1, p2)* with *p1,p2* ∈ *portion* **Do**:
    a. flip edge *(p1, p2)*

Algorithm 3: Deriving an Efficient PM from a PN:
deriveEfficientModel(PN)
Input: A PN, *PN* Output: A PM

1. *M* = empty PM
2. *M.add_nodes(PN.assigned_process.functions)*
3. **For Each** edge *(p1, p2)* **In** *PN.social_edges* **Do**:
    a. **For Each** (*n1, n2*) ∈ *(p1.assigned_nodes, p2.assigned_nodes)* **Do**:
        i. **If** there is at least one edge (*ip1, ip2*) in (*n1.involved_performers, n2.involved_performers*) **Then**:
            1. *M.add_edge(n1, n2)*
4. *M.delete_unconnected_nodes()*
5. **return** *M*

Algorithm 4: Fitness of a PN: *fitness(PN, Models)*
Input: A PN, *PN* Output: A numeric fitness

1. **return** $\sum efficiency(PN, m) * (1 - dens(PN))$ for each *m* in *Models*

Algorithm 5: Efficient PN Generation around a PM Based on [9]: *generatePN(PM)*
Input: A PM, *PM* Output: A PN

1. let *PN* be a random social network
2. assign *PN.performers* to random capabilities from *PM*
3. assign *PN.performers* to random process functions from *PM*
4. simulate *PN* around *PM*
5. **Repeat** 1-4 until *PN.efficiency* reaches a local optimum after an adequate number of iterations
6. **return** *PN*

4 Validation of Concept

4.1 Experimental Design

The validation of concept elucidates its potential and applicability. Herby, the concept approach is validated to be able to reproduce its results (reliability), to explain the increase of result quality (intern validity) and to produce a result that is generalizable/transferable (extern validity) [8]. For that purpose, we implemented our approach within the "Refmod-miner" framework which is a prototypical software platform (http://refmod-miner.dfki.de/). All algorithms described in the approach section were implemented in Java, the used network generators and SNA algorithms in section 2.1 were taken from the Python library NetworkX 1.11. The hardware configuration on which the implementation of our approach was executed involves an Intel(R) Core(TM) i7-3610QM CPU @ 2.30GHz and 8GB of RAM.

We provide three evaluation scenarios in which we demonstrate the reliability, intern validity and extern validity of our approach. The scenarios rely on three different real-world knowledge-based domains and two different languages, German and English. The three scenarios cover S1: 5 knowledge transfer PMs from knowledge management in outsourcing relationships and knowledge progress control [34], S2: 10 PMs created during the workshop for modelling in higher education (MoHoL 2016) [35]. S3: 7 PM variants/solutions about business trip admission for business informatics assignments in exams at a German university [36]. For each scenario, also a gold RM is provided, made by a domain expert, whose execution comes to an exemplary result comparing to the models in its scenario. For example, in S2, at the end, a customer is informed and in S3, a car for a business trip is ordered. PM properties, such as the sum, minimum value, first, second and third quartile and the maximum value for the number of nodes, start nodes, number of process edges and graph density, are presented in table 1.

Table 1. Overview over PM metrics: sum, minimum, maximum and quartiles

S1|**S2**|S3	sum	min	q25	median	q75	max
nodes	194|**187**|264	18|**15**|19	22|**17**|36	28.5|**18.5**|41	35|**19**|43	62|**26**|43
start nodes	6|**9**|11	1|**1**|1	1|**1**|1	1|**1**|2	1|**1**|2	1|**1**|2
edges	221|**200**|280	23|**17**|20	23|**18**|38	30|**20.5**|44	39|**21**|45	76|**25**|46
density in %	26|**62**|21	2|**4**|3	3|**6**|3	4|**6**|3	5|**7**|3	8|**8**|6

As described in the approach, our prototypical concept implementation runs *developReferenceModel(Models)* 1000 times. The mutation rate was set to 0.2 as this rate indicated significantly better results than 0.1, 0.3 and 0.4 in the first 10 runs. For each scenario corpus, each evaluation of PNs is repeated 100 times, and records these topological properties of the evolving individuals in every iteration: *num_per* (number of performers), *md* (average performer degree), *CC* (average clustering coefficient), *dens* (the PN density), *minimum_fitness* (defined in *developReferenceModel* as the fitness of the best PN for the input PMs, respectively the best initial individual (remains constant during the algorithm run), *cur_fitness* (fitness of the actual evolving individual), *best_fitness* (fitness of the present best individual). The fitness of the respective gold RM *gold_fitness* is measured to ensure that the gold RMs are comparable to our generated models. The fitness of a gold RM is evaluated by the same procedure as the minimum fitness is calculated, namely by the fitness sum of its most efficient PN towards all input PMs.

4.2 Evaluation

The runtime of *developReferenceModel* took on average 15 seconds per run. Runs in scenario S2 took on average 20% longer than the others and runs in S3 took on average 7% longer than S1. One iteration took between 0.25 and 2 seconds. The fitness of evolving individuals increases with the number of iterations in one run for reaching *cur_fitness (#iterations*) during the algorithm ($p < 0.001$) in all scenarios which can be observed in figure 2. Each point stands for a run of the PN development algorithm (the respective scenario that was run is marked with its own colour).

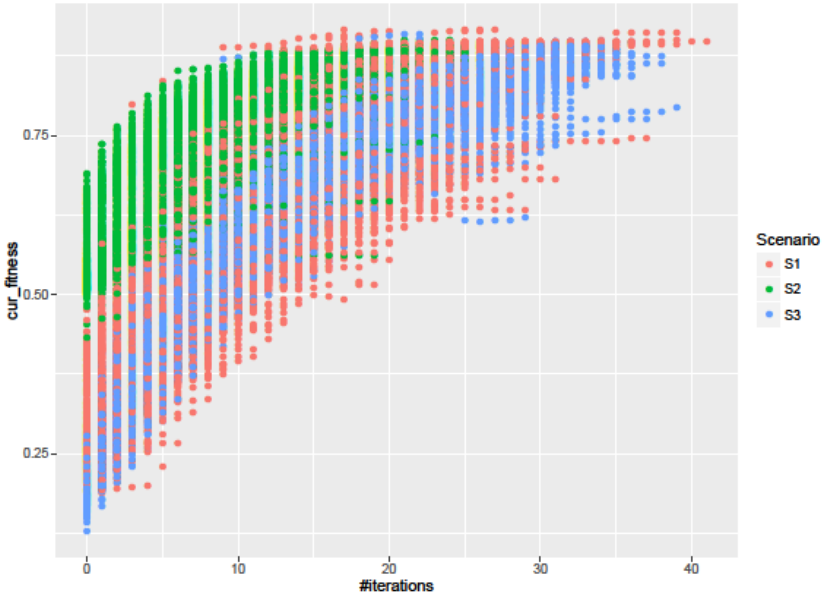

Figure 2. *#iterations* vs *cur_fitness* for all scenarios

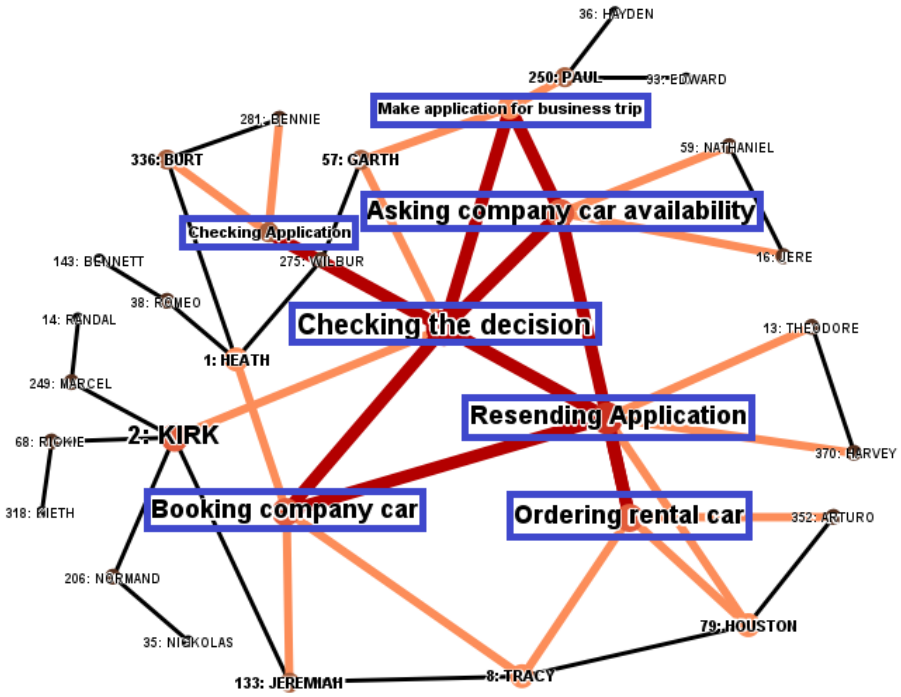

Figure 3. Reference PN with its derived RM for S3

Table 2. Descriptives for *best_fitness* over all runs with smallest value (Min), arithmetic mean (Mean), standard deviation (SD) and highest value (Max)

Scenario	Min	Mean	SD	Max
S1	0.84	0.89	0.01	0.92
S2	0.84	0.88	0.01	0.90
S3	0.84	0.88	0.01	0.91

Table 3. Average efficiency over all algorithm runs with standard deviation in brackets

Scenario	*best_fitness / minimum_fitness*	*best_fitness / gold_fitness*	*gold_fitness / minimum_fitness*
S1	4.67 (0.07)	1.19 (0.04)	3.92 (0.13)
S2	1.76 (0.02)	1.56 (0.04)	1.13 (0.03)
S3	6.01 (0.10)	2.58 (0.12)	2.33 (0.10)

Table 4. Correlations between respective variable and *cur_fitness*, $p < 0.01$

Scenario	*#iterations*	*num_per*	*md*	*CC*	*dens*
S1	0.86	0.86	-0.90	0.86	-0.88
S2	0.80	0.80	-0.82	0.77	-0.79
S3	0.92	0.93	-0.92	0.86	-0.88

In no run, more than 42 iterations were needed to reach convergence. As we can see in table 2, the mean best reached fitness lies nearer to its maximum than to its minimum. The standard deviation of *best_fitness* amounts to 0.01 for all scenarios. All PNs generated from the scenario respective gold RMs have a very low *num_per*, *CC* and *dens* with a simultaneously high efficiency and fitness over all input PMs. Figure 3 presents the fittest PN together with its derived PM, which we see as RM for all input PMs in S3. The figure contains process functions and process edges of the RM, performers of the reference PN and the related functional as well as social edges. All edges are undirected indicating mutual relations. Nodes with numbers are generated performers and the others with blue boxes are process functions. Black curves are social edges, red curves are process edges and orange curves are functional edges. Size and colour intensity of nodes and edges indicate the strength of their degrees. For reasons of clarity and comprehensibility only performers and process functions with a degree greater than 2 are plotted. In table 3, the average efficiency of all algorithm runs is presented. The average efficiency is the ratio between *best_fitness* and *minimum_fitness* of a run, averaged over all runs. The same applies for the relations between *best_fitness* and *gold_fitness* and *gold_fitness* and *minimum_fitness*. As an example, the fitness of the most efficient evolved PN in S3 was 2.58 higher than the respective gold PN and 6.01 times higher than the highest fitness of all individuals from the initial population (*minimum_fitness*). The fitness of the gold PN was 2.33 times greater than the minimum fitness. The relation between the fitness of the gold PN and the best initial individual was 2.33. Each standard deviation is significantly lower than a tenth of the corresponding average relation which speaks for a sufficient number of algorithm runs for this evaluation.

4.3 Discussion

The mutation rate of 0.2 reached a fitness convergence in all scenarios. Maximal 5 runs were needed to find an acceptable individual. The degree distributions of all efficient PNs around the gold RM and the best evolved PN, for all scenarios show the existence of hubs. Also, those PNs exhibit a significant but low clustering and density. The existence of hubs is an evidence for hierarchy which means that teams/clusters have single members that are much more central (between more people) than others [22]. This finding seems to be a condition for efficiency which is also confirmed by [9] for knowledge working processes. The number of performers correlates with the fitness over many iterations which means that more performers make the evolving PN more efficiently but this effect is compensated by the negative influence of the mean degree. All correlations in table 4 are stable over the scenarios in the meaning of their equal sign. The topological PN properties density, clustering and mean degree seem to be fitness drivers. For the approach's reliability pleads this stability after 1000 repetitions of the algorithm and of the repeated generation/evaluation of each PN.

According to our experimental design, we see the intern validity as confirmed by the significant fitness increase towards the minimum fitness for all scenarios over all runs (h0: *cur_fitness* < *minimum_fitness*; $p < 0.001$). Also there is a significant

correlation between the number of iterations and the fitness ($p < 0.001$). Our algorithm so has an ascertainable influence on the evolvement of better individuals. For the extern validity, we tested the significant fitness increase towards the fitness of the gold reference PNs for all scenarios over all runs (h0: *cur_fitness* < gold_fitness; $p < 0.001$). The relation between the gold reference PN and the minimum fitness was always significantly greater than 1.0. That is an assertive indicator for our algorithm to be able to generate efficient reference PNs for various scenarios. All gold RMs were evaluated to be highly efficient and fit. This speaks for the validity of our approach because arbitrary models out of the initial population are significantly less efficient (h0: *fitness(generatePN(m), Models)* > *fitness(generatePN (gold_PM), Models)* for each PM *m* in the input PMs *Models* and the respective gold RM; $p < 0.01$). That means that the generated PN around the gold RM is efficient for all input PMs. Considering the low density and the random assignment of capabilities to performers in an efficient generated PN, its performers have only the most critical capabilities to work on the most critical process functions over all input PMs. This also means that the gold RMs' efficiency can be compared to other RMs for the input PMs which makes our fitness function a valid indicator for the quality of a RM.

The advantage of our approach is that the RMs can be generated valid over different domains. The proposed PN/RM combination describes a minimum topology of performers and their assigned process functions that is efficient towards the set of given input PMs. This proposed PN and RM can be adapted for specific stated requirements such as pre-given teams or pre-assigned performers to certain process functions. For a real environment, their efficiency comparing to other team/hierarchy constellations can then be simulated and evaluated. The RM in figure 3, as an example, indicates that the work flow between "Booking company car", "Ordering rental car" and "Checking the decision" is most critical for the efficiency of all process variants in S3 as they have the highest degrees of all process functions which makes them central in the RM. These process functions lie on critical paths in most process variants in S3, in the meaning of paths that reach from start nodes to end nodes and lie on many other paths at the same time. For that reason, most of all hub performers, such as "Kirk" "Jeremiah" and "Heath", were placed to work with their subordinated teams at this process functions. This can be interpreted as a recommendation or reference for a modeller to focus on needed capabilities for this process region when positioning real personal, e.g. at checking the decision for booking a car.

Limitations: Our approach focusing only on PN/PM topology is quite abstract and based on simplified assumptions about real processes and organizations. Organizations in our approach only consist of a set of performer networks and PMs. Their execution environment, social behaviour, resource allocation, communication- and production/processing capacities are not considered. In order to demonstrate the potential of our approach, the implemented algorithm produces quickly an acceptable result but will hardly reach a global optimum. For achieving a much better result, the number of performers should be reduced nearly to the number of needed performers in the gold RMs. An adapted, organization-specified implementation that considers

the environment, the concrete performer capabilities and restrictions for their process assignment will be imperative for our approach to be utilized by practitioners.

5 Conclusion

In this paper we introduce a new concept for the inductive development of reference process models. A social perspective for matching the process flow is applied, rather than a traditional label matching which is an inexact and subjective approach. For a given set of input PMs, a reference process model is developed, in a few seconds of runtime, by including all process functions that are minimum requirements for the resulting model to be efficient. The efficiency is measured by the time that simulated performers need to complete the process. Three evaluation scenarios are provided to evaluate our approach. The evaluation indicates that the generated reference process models are at least as efficient as the input PMs and as a RM designed by an expert. Our results confirm the potential of our approach as they confirm its extern validity.

From a theoretical point of view that means that the efficiency of RMs designed by experts can be compared to our developed RMs which makes our fitness function a valid indicator for the quality of a RM. This in turn implies social collaboration to be an important facet for reference modelling. Our approach can be tailored to concrete organizations and processes. Practitioners take advantage of pre-selecting efficient sub corpora out of many models and identifying maybe invisible lead performers / critical junctures, in contrast to the formal structure, constituting efficient structures of co-worker ship around process models.

In future works, we want to provide a method to evaluate the quality of reference process models based on this approach. Also we will evaluate event logs of the execution of business processes to add a time/cost component to our fitness function for developing reference process models.

References

1. Becker, J. and Meise, V.: Strategy and Organizational Frame, Process Management. A Guide for the Design of Business Processes, J. Becker, M. Kugeler and M. Rosemann (eds.), Springer, 2011.
2. Fettke, P., Loos, P.: Perspectives on Reference Modeling. In: Fettke, P., Loos, P. (eds.) Reference Modeling for Business Systems Analysis, pp. 1-20. Idea Group, Hershey, PA., 2007.
3. Walter, J., Fettke, P., Loos, P.: How to Identify and Design Successful Business Process Models: An Inductive Method. In: Becker, J., Matzner, M. (eds.) Promoting Business Process Management Excellence in Russia - Proceedings and Report of the PropelleR 2012 Workshop, pp. 89-96. Moscow, Russia, 2013.
4. Becker, J., Schütte, R.: A Reference Model for Retail Enterprises. In: Fettke, P., Loos, P. (eds.) Reference Modeling for Business Systems Analysis, pp. 182-205. Idea Group, Hershey, PA, 2007.

5. Thaler, T., Hake, P., Fettke, P., Loos, P.: Evaluating the Evaluation of Process Matching Techniques. In: Kundisch, D., Suhl, L., Beckmann, L. (eds.) Tagungsband Multikonferenz Wirtschaftsinformatik 2014, MKWI-2014, pp. 1600-1612. Paderborn, Germany, 2014
6. Wasserman, S. and Faust, K., Social network analysis: Methods and applications, volume 8. Cambridge university press, 1994.
7. Martens, A., Fettke, P., Loos, P.: A Genetic Algorithm for the Inductive Derivation of Reference Models Using Minimal Graph-Edit Distance Applied to Real-World Business Process Data. In: Kundisch, D., Suhl, L., Beckmann, L. (eds.) Tagungsband Multikonferenz Wirtschaftsinformatik 2014, MKWI-2014, pp. 1613-1626. Paderborn, Germany, 2014.
8. Campbell, DT., and Stanley, JC.: Experimental and quasi-experimental designs for research. Ravenio Books, London, 2015.
9. Sonntag, A., Fettke, P.: Efficiency Of Generated Performer Networks In Collaborative Business Process Models. In: IEEE Conference on Business Informatics (CBI). Conference on Business Informatics (CBI-16), August 29 - September 1, Paris, France, IEEE, 2016.
10. Beyer, H-G.: The theory of evolution strategies, Springer Science & Business Media, 2013.
11. Holme, P. and Kim, B. J.: Growing scale-free networks with tunable clustering, Physical Review E, 65(2): 026107, 2002.
12. Niehaves, B. and Plattfaut, R.: Collaborative business process management: status quo and quo vadis, Business Process Management Journal, 17(3): 384–402, 2011.
13. Beyer, HG, and Schwefel, HP.: Evolution strategies–A comprehensive introduction. Natural computing 1.1: 3-52, 2002.
14. Hevner, A., March, S., Park, J. and Ram, S.: Design science in information systems research, MIS Quarterly, Vol. 28 No. 1, pp. 75–105, 2004.
15. Cross, R., Borgatti, S. P., and Parker, A.: Making invisible work visible: Using social network analysis to support strategic collaboration, California management review, 44(2):25–46, 2002.
16. Garey, M.R., Johnson, D.S.: Computers and Intractability: a Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.
17. Scheer, AW, and Nüttgens, M.: ARIS architecture and reference models for business process management. Business Process Management. Springer Berlin Heidelberg, 376-389, 2000.
18. Fettke, P. and Loos, P.: Multiperspective evaluation of reference models–towards a framework. International Conference on Conceptual Modeling. Springer Berlin Heidelberg, 2003.
19. Barabási, AL.: Network Science. Cambridge University Press. Retrieved 25 May, 2015.
20. Watts, DJ.: Six degrees: The science of a connected age, WW Norton and Company, New York, 2004.
21. Leskovec, J., Kleinberg, J., and Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations, In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pages 177–187, ACM, 2005.
22. Barabási, A.-L. and Albert, R.: Emergence of scaling in random networks, science, 286(5439):509–512, 1999.
23. Johnson-Cramer, M. E., Parise, S., and Cross, R. L.: Managing change through networks and values, California Management Review, 49(3):85–109, 2007.

24. Cross, R., Borgatti, S. P., and Parker, A.: Making invisible work visible: Using social network analysis to support strategic collaboration, California management review, 44(2):25–46, 2002.
25. Gottschalk, F., Van Der Aalst, W., Jansen-Vullers, M.: Mining reference process models and their configurations. In: Meersman, R., Tari, Z., Herrero, P. (eds.) On the Move to Meaningful Internet Systems: OTM 2008 Workshops. Lecture Notes in Computer Science, vol. 5333, pp. 263–272. Springer, Berlin, 2008.
26. Ardalani, P., Houy, C., Fettke, P. and Loos, P.: Towards a Minimal Cost of Change Approach for Inductive Reference Model Development, Proceedings of the 21st European Conference on Information Systems, AIS, Utrecht, 2013.
27. Li, C., Reichert, M. and Wombacher, A.: Discovering Reference Models by Mining Process Variants Using a Heuristic Approach, in Dayal, U., Eder, J., Koehler, J. and Reijers, H. (Eds.), Business Process Management: 7th International Conference, BPM 2009, Ulm, Germany, September 8-10, 2009. Proceedings, Vol. 5701, Springer, Berlin, Heidelberg, pp. 344–362, 2009.
28. Rehse, JR., Fettke, P. and Loos, P.: A graph-theoretic method for the inductive development of reference process models, Software & Systems Modeling, 2015.
29. Rehse, JR.; Fettke, P.; Peter Loos, P.: An Execution-Semantic Approach to Inductive Reference Models Development, in: 24th European Conference for Information Systems (ECIS-16), June 12-15, Istanbul, Turkey, Association for Information Systems (AIS), 2016.
30. Yahya, B.N., Wu, J.-Z. and Bae, H.: Generation of Business Process Reference Model Considering Multiple Objectives, Industrial Engineeering & Management Systems, Vol. 11 No. 3, pp. 233–240, 2012.
31. Yahya, B.N. and Bae, H.: Generating Reference Business Process Model Using Heuristic Approach Based on Activity Proximity, Intelligent Decision Technologies, Springer, pp. 469–478, 2011.
32. Martens, A., Fettke, P. and Loos, P.: Inductive Development of Reference Models Based on Factor Analysis, in Thomas, O. and Teuteberg, F. (Eds.), Proceedings Der 12. Internationalen Tagung Wirtschaftsinformatik (WI 2015), Vol. 12, Universität Osnabrück, Osnabrück, Osnabrück, Germany, pp. 438 – 452, 2015.
33. Ling, J. and Zhang, L.: Generating Hierarchical Reference Process Model Using Fragments Clustering, Asia-Pacific Software Engineering Conference (APSEC). IEEE, 2015.
34. Novara, C., and Schwabe, G.: Wissensmanagement in Outsourcingbeziehungen und Wissenskulturfortschrittskontrolle, Chur Schweiz, 2006.
35. Workshop for modeling in higher education (MoHoL 2016), http://butler.aifb.kit.edu/MoHoL/?page_id=62 (Accessed: 21.10.2016)
36. Repository of business informatics exams solutions, http://rmm.dfki.de/index.php?site=repository&file=Exams&source=repo (Accessed: 21.10.2016)