Design it like Darwin 2.0 - An Evolutionary Algorithm for Automatic Process Redesign

by

Johannes Seyfried, Tobias Ruby, Maximilian Röglinger, Jonas Manderscheid[1]

in: Proceedings of the BPM Demo Track 2017, 2017, p. 1-5

[1] Hilti Befestigungstechnik AG

Design it like Darwin 2.0

An Evolutionary Algorithm for Automatic Process Redesign

Jonas Manderscheid[1], Maximilian Röglinger[2], Tobias Ruby[1], Johannes Seyfried[2]

[1]FIM Research Center, University of Augsburg, Germany
{jonas.manderscheid, tobias.ruby}@fim-rc.de
[2] Project Group Business & Information Systems Engineering of the Fraunhofer FIT, Germany
{maximilian.roeglinger, johannes.seyfried}@fim-rc.de

Abstract. Process redesign is an important and valuable phase of the business process management (BPM) lifecycle. However, human creativity and objectiveness regarding continuous redesign initiatives are limited and biased. To overcome these limitations, we propose computational support based on evolutionary algorithms. Our software tool extends a formerly published proof of concept. Novelties have been introduced by including a new data structure, new mutation and crossover operators as well as an extended evaluation of unambiguous process designs explicitly considering time objectives. Finally, the tool provides new computation process (re-) design support to the BPM community.

Keywords: business process management, process redesign, computational intelligence, evolutionary algorithm

1 Background and Significance to the BPM field

Business process management (BPM) is an acknowledged way to facilitate value-creating process improvement activities [1]. Nevertheless, process (re-) design mainly relies on human intuition limiting the solution space [2]. The practical toolkit for process (re-) design still lacks computational support [3], even though algorithms can outperform humans in creating and benchmarking processes due to their speed and neutrality [4]. Consequently, computational intelligence (CI) in terms of machine learning abilities has been introduced to business process (re-) design in the mid-nineteens. After initial process automation approaches [5], CI's rising popularity meanwhile facilitated computational tool support for process (re-) design [6]. The tool portfolio ranges from semi-automatic approaches to the successfully application of evolutionary algorithms (EA), as part of CI, for automatic process (re-) design [7,8].

EAs describe heuristics tightly related to Darwin's approach of natural selection, known as *survival of the fittest* [9], as the idea behind EA is similar to the BPM lifecycle: developing improved generations of business processes. In doing so, EA start with a population of known and/or randomly created genomes (i.e., process designs). EA operators then mimic the BPM lifecycle phases and foster the evolution of this population by creating new generations of genomes based on the most valuable existing ones.

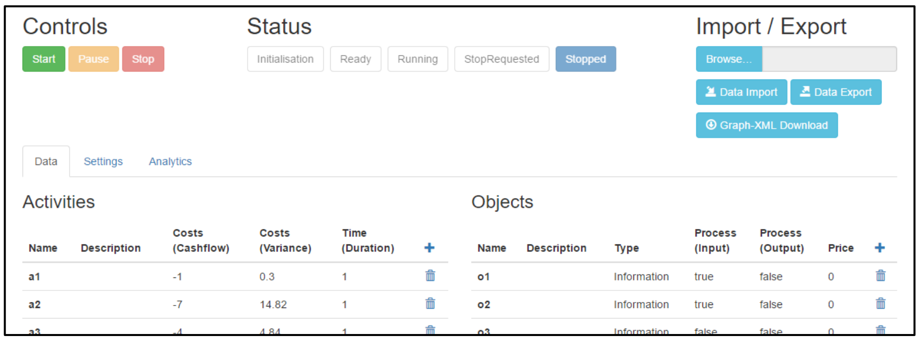
The mutation operator randomly replaces sub-processes with new gateways and/or tasks, whereas the crossover operator combines proven sub-processes of existing genomes (see Sect. 2). The valuation of genomes is done by a fitness function depending on predefined criteria. The evolution cycle repeats until it results in a (potentially local) optimum or it is terminated manually.

Afflerbach et al. [10] provide the first EA application that allows for exclusive splits and covers elementary process design elements (i.e., tasks, input/output objects, and sequence flows). In addition, it aggregated multiple dimensions of process performance to the main economic factors of cash flows and the risk considering time, quality, and flexibility [11]. They confirm the usefulness of their EA design in a proof of concept. However, the proof of concept still lacks scalability, platform- and vendor independence, well-defined interfaces, visualization and analysis functionalities as well as a direct integration of performance effects. We therefore present a stand-alone process (re-) design tool *Design it like Darwin 2.0* that addresses many of these shortcomings. The tool enables researchers and practitioners to create new process designs based on any set of tasks (e.g., gathered by process mining). Particularly, the import/export feature provides well-defined interfaces as further step towards a highly automated BPM lifecycle.

2 Design it like Darwin 2.0 – The Tool

Regarding the tool presentation, we start with an architectural overview followed by a description of the significantly enhanced backend features. Finally, we present the new analytics and visualization functionalities.

The tool is a dynamic single page C# web application that also implements an application programming interface (API). Up to 19 **settings** allow for the customization of the EA itself (e.g., the maximum number of generations or population size) as well as process improvement goals (e.g., weights for the performance dimensions and the decision-makers' risk attitude). **Input data** regarding tasks (i.e., name, description, expected cash flow and variance, and execution time), input/output objects, input/output dependencies, process attributes, and attribute coverage can be inserted via user interface (see Fig. 1) or by a JSON file upload.



Activities

Name	Description	Costs (Cashflow)	Costs (Variance)	Time (Duration)	+
a1		-1	0.3	1	🗑
a2		-7	14.82	1	🗑
a3		-4	4.84	1	🗑

Objects

Name	Description	Type	Process (Input)	Process (Output)	Price	+
o1		Information	true	false	0	🗑
o2		Information	true	false	0	🗑
o3		Information	false	false	0	🗑

Fig. 1. Tool user interface (Data tab)

For the backend calculation, we implemented the following key functionalities, which enhance the existing proof of concept of Afflerbach et al. [10]:

Greedy Algorithm. We apply a greedy algorithm to generate new genomes while randomly combining tasks according to their input/output objects, dependencies, and gateways. This choice significantly influences the performance. The resulting genomes are feasible (i.e., process attributes constraints and data input/output dependencies are considered), but rarely optimal in the first iteration. Besides, we can force the greedy algorithm to create only feasible genomes to facilitate an automated initial process design.

Tree structure. We present processes as directed trees (see Fig. 2). Tree nodes correspond to parallel, sequential, and exclusive gateways, where leaves correspond to tasks. Thereby, we explicitly consider the semantics of BPMN gateways and their semantics in EA operations (mutation and crossover) and, thereby, extend the overall solution space compared to the existing proof of concept.

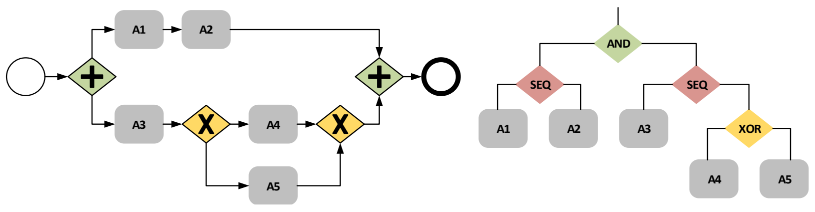


Fig. 2. Sample BPMN process transformation into a tree

Genome Operators. With the introduction of the directed tree, we also further developed the implemented EA operators according to Poli et al. [12]. The crossover operator randomly switches two nodes of two genomes, while the mutation operator generates new subtrees using the greedy algorithm to replace existing ones. We also allow for limiting the tree depth to avoid unnecessarily complex process designs with multiply duplicated gateways and/or tasks not contributing to better solutions.

Varying Crossover-/Mutation-Rates. As varying crossover and/or mutation rates is critical to the success of EAs, we start with a high mutation probability and a low crossover probability that will decrease or increase respectively by $1/n$ per generation [13], where n represents the number of generations. Thereby, we ensure a broad exploration of the solution space at the beginning and a narrow solution space exploitation at the end, fostering the convergence of the EA.

Multi-criteria Evaluation. We explicitly model the information related to the process performance dimension time (i.e., fixed completion time per task) to foster the use of exclusive and parallel gateways. We finally calculate the genomes' fitness as an individually weighted average of the task-related cash flows and time estimations based on user preferences [10]. This calculation covers the process performance dimensions time and cost.

Pain factor. We widened the solution space by infeasible (i.e., violated input-output-constraints) and overly complex (e.g., redundant gateways or parallel gateways with only one path) process designs. As the best solution is close to the boundary between the solution space of feasible and infeasible designs, we tackle the best solution from both sides. To cope with such designs, we introduced a pain factor as a negative adjustment in their fitness, punishing infeasible and overly complex solutions. Besides, we ensure at least one feasible solution per generation through the greedy algorithm (see above) or the elitist selection [12].

For the presentation of the results, we introduced new **analytics and visualization** features. The analytics tab displays the development of several performance indicators in real-time (e.g., the fitness development, the percentage of feasible process designs, or the evaluation runtime, see Fig. 3). Besides, the analytics tab always presents the best process design so far as BPMN graph and provides a BPMN 2.0 XML file download for further processing.

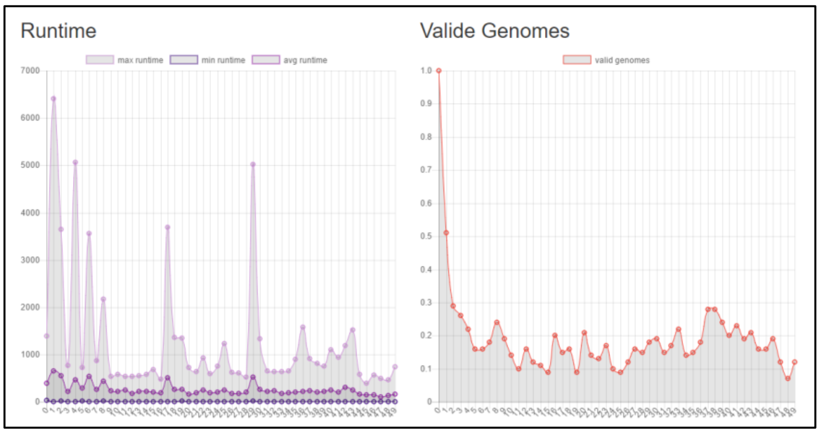


Fig. 3. Tool user interface (Analytics tab)

3 Maturity and Future Work

Design it like Darwin 2.0 is a ready-to-use fully automated process (re-) designer applicable to academic and industrial contexts. We invite the BPM community to challenge their own process design ideas.

From an evaluation perspective, we compared the enhanced algorithm to the proof of concept in [10] based on the same travel agent process from a modified real-life scenario. The greedy algorithm generates populations with thousands of genomes as recommended for EAs based on tree structures within seconds. As the best solution is close to the boundary, the example process already reveals the value of the extended solution space (i.e., feasible and infeasible designs). Even though the overall average of feasible genomes is about 10 to 15 percent per generation owing to infeasible process designs, the tool identifies the optimal solution more quickly than the initial proof of concept. Therefore, our proposed enhancements also have significant effects on performance.

To further develop our tool, we plan to extend the fraction of supported BPMN elements (i.e., all BPMN gateways and events) and, therefore, the complexity of supported processes as a next step. Besides, we intend to support process with loops of recurring tasks. To do so, we must upgrade the underlying data structure to a directed graph and implement corresponding EA operators according to Walker et al. [14]. We expect particular challenges when extending our tool to support also declarative process models.

The tool, installation instructions, and example data is available at **https://github.com/rubytobi/Design-it-like-Darwin-2.0**. Additionally, we provide a screencast showing the basic workflow with the application.

The tool and its source code is available under MIT License.

References

1. Dabaghkashani, A.: A Success Model for Business Process Management Implementation. International Journal of Information and Electronics Engineering **2**(2), 725-729 (2012).
2. Sharp, A., McDermott, P.: Workflow modeling. Tools for process improvement and applications development / Alec Sharp, Patrick McDermott, 2nd edn. Artech House, Boston, London (2009).
3. Zellner, G.: A structured evaluation of business process improvement approaches. Business Process Management Journal **17**(2), 203-237 (2011).
4. van der Aalst, W.M.P.: Business Process Management. A Comprehensive Survey. International Scholarly Research Notices Software Engineering **2013**, 1-37 (2013).
5. Hammer, M., Champy, J.: Reengineering the corporation: A manifesto for business revolution. Business Horizons **36**(5), 90-91 (1994).
6. Bernstein, A., Klein, M., Malone, T.W.: The process recombinator. A tool for generating new business process ideas. In: Proceedings of Information Systems, 178–192 (1999).
7. Vergidis, K., Saxena, D., Tiwari, A.: An evolutionary multi-objective framework for business process optimisation. Applied Soft Computing **12**(8), 2638-2653 (2012).
8. Low, W.Z., Weerdt, J. de, Wynn, M.T., ter Hofstede, A.H.M., van der Aalst, W.M.P., vanden Broucke, S.: Perturbing event logs to identify cost reduction opportunities. A genetic algorithm-based approach. In: Proceedings of Evolutionary Computation, 2428–2435 (2014).
9. Fogel, D.B.: What is evolutionary computation? IEEE Spectrum (2000).
10. Afflerbach, P., Hohendorf, M., Manderscheid, J.: Design it like Darwin. A value-based application of evolutionary algorithms for proper and unambiguous business process redesign. Journal of Information Systems Frontiers (2016).
11. Limam Mansar, S., Reijers, H.A.: Best practices in business process redesign. Use and impact. Business Process Management Journal **13**(2), 193-213 (2007).
12. Poli, R., Langdon, W.B., McPhee, N.F., Koza, J.R.: A field guide to genetic programming. Lulu Press (2008).
13. Lin, W.-Y., Lee, W.-Y., Hong, T.-P.: Adapting crossover and mutation rates in genetic algorithms. Journal of Information Science and Engineering **19**(5), 889–903 (2003).
14. Walker, J.A., Miller, J.F.: The Automatic Acquisition, Evolution and Reuse of Modules in Cartesian Genetic Programming. Transactions of Evolutionary Computation **12**(4), 397-417 (2008).