Technical University of Denmark

DTU

Evolutionary optimization of production materials workflow processes

Herbert, Luke Thomas; Herbert-Hansen, Zaza Nadja Lee; Jacobsen, Peter; Cunha, Pedro

DTU Library
Technical Information Center of Denmark

8th International Conference on Digital Enterprise Technology - DET 2014 – "Disruptive Innovation in Manufacturing Engineering towards the 4th Industrial Revolution

Evolutionary optimization of production materials workflow processes

Luke Herbert[a]*, Zaza Nadja Lee Hansen[a], Peter Jacobsen[a], Pedro Cunha[b]

[a]The technical University of Denmark, DTU Compute and DTU Management, Anker Engelunds Vej 1, 2800 Kongens Lyngby, Denmark
[b]Instituto Politécnico de Setúbal, Rua da Estefanilha, 2910-761 Setúbal, Portugal

* Corresponding author. Tel.: +45-45255290; fax: +45-45933435. E-mail address: lthhe@dtu.dk

Abstract

We present an evolutionary optimisation technique for stochastic production processes, which is able to find improved production materials workflow processes with respect to arbitrary combinations of numerical quantities associated with the production process. Working from a core fragment of the BPMN language, we employ an evolutionary algorithm where stochastic model checking is used as a fitness function to determine the degree of improvement of candidate processes derived from the original process through mutation and cross-over operations. We illustrate this technique using a case study where a baked goods company seeks to improve production time while simultaneously minimising the cost and use of resources.

Keywords: Evolutionary algorithm; Stocastic BPMN;Pproduction optimisation

1. Introduction

The first method for documenting process flow can be traced back to Frank Gilbreth's seminal 1921 paper (Gilbreth, 1921). The later development of the ideas originating in his paper have been crucial in the development of today's concept of Business Process Modelling (BPM), which is concerned with mapping workflow processes, for example in production, to enable analysis and improvement of organisational efficiency and quality. However, modern enterprises, in particular those involved in producing highly engineered products or addressing dynamic customer needs, are challenged to form complex networks in which the need to adapt to a constantly changing environment is crucial.

Disruptive technology or changing customer demands can allow for radically new ways of doing business. Being the first to fully realise the beneficial possibilities of these advances and adopting the radically improved production and business processes which they allow, can be the key to a competitive advantage. For example, Toyota is widely viewed as being the first to truly realise the revolutionary benefits of IT based inventory control and the consequent Just-In-Time production workflow processes it enabled (Shingo, 1981), however, the technologies which enabled this advance were in place several decades before. This example illustrates that business and production workflow processes are interlinked and interdependent. Therefore the same approach and tools are used when analysing and optimising production workflow processes as business workflow processes.

A production process is for example cutting, forming or molding a product or conducting quality control measures. Developing production and business processes is today predominantly an activity in which tools are used to draw the process maps. The processes are analysed by hand and improved configurations are found by a process of trial and error, often taking too much time to arrive at an optimal practice due to the learning experience involved.

In this paper we will describe how an evolutionary algorithm can be used, combined with stochastic model checking, to optimise a stochastic process in production by employing the prototype software tool SBOAT. The

applicability of the method is shown through an example from the Danish baked goods industry.

2. Related work

In previous work we have developed methods that allow for models of production and business processes expressed in the Business Process Model and Notation (BPMN) modelling language to be analysed for properties described using an extended form of the temporal logic Probabilistic Computation Tree Logic (PCTL) (Aziz et al., 1995). This allows the precise calculation of the timing, occurrence and ordering of events, transient and steady-state probabilities, and reward-based properties. Here we build upon these developments to allow for optimisation of production and business processes where optimisation objectives may be broadly defined, related for example with the use of resources, cost and time.

2.1. Stochastic model checking

The goal of this work is to transform a BPMN model into a Markov decision process (White, 1993) (MDP) which is amenable to formal state space analysis. These states represent possible configurations of the modelled system with probabilistic state transitions being combined with non-deterministic choices between several discrete probability distributions over successor states. Model checking allows for the efficient exploration of the entirety of this space with a temporal logic employed to select sets of states of interest, and offers the possibility of verifying many properties of a system. In this paper we will specifically use this capability to select sets of paths through the state space that represent different strategies; each path is then checked to ensure that given safety criteria are observed and the values of rewards of interest are computed.

2.2. Business process and notation

The Business Process Model and Notation (BPMN) language (OOG, 2011) is a widely adopted graphical notation for specifying workflow processes. The semantics and pragmatics of BPMN are, however, only informally defined in the relevant standards (OOG, 2011), thus leaving a number of questions open to interpretation. There are essentially only two fundamental types of object, nodes and flows, and in this work only a small subset of BPMN, often known as the core subset, is used. This consists of the eight elements found to be the most commonly used in a large survey of real-world BPMN usage (Muehlen & Recker, 2008).The graphical elements of core BPMN are shown in Figure 1 and described in Definition1 below. It should be noted that by combining several Core BPMN elements any element of the complete BPMN language can be simulated, even inclusive gateways (Christiansen et. al., 2011).

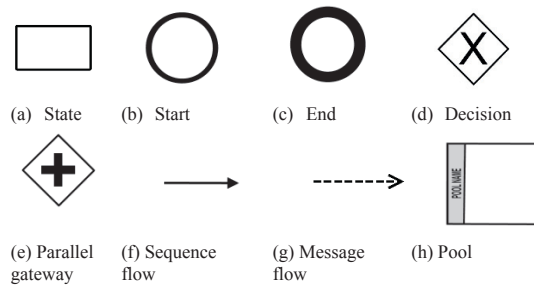BPMN modelling involves composing a number of elements into a business process diagram (BPD).

Fig. 1. Core BPMN elements.

Definition 1 (Stochastic Core BPD). A Stochastic Core BPD is a tuple $BPD = (N, F, P, pool, L, lab, P)$ where $N \subseteq T \cup E \cup G$ is a set of nodes composed of the following disjoint sets:

- Tasks **T** are the basic actions done as part of a given workflow process, e.g. "sending a letter" or "putting sprinkles on a cake".
- Events $E \subseteq E^S \cup E^E$ where the disjoint sets E^S and E^E respectively represent start and end events.
- Gateways $G \subseteq G^D \cup G^F \cup G^M$, where the disjoint sets G^D, G^F and G^M respectively represent exclusive decision gateways, parallel fork gateways and parallel merge gateways.

$F \subseteq S \cup M$ *is a set of flow relations, where sequence flows* $S \subseteq N \times N$ *relate nodes to each other and* $M \subseteq T \times G^M$ *is a relation between tasks and parallel merge gateways.* $P \cup \wp(N)$ *is a set of disjoint pools and pool:* $pool: N \rightarrow P$ *assigns nodes to a pool* $p \in P$ *is a set of unique labels and* $lab: F \rightarrow L$ *is a labelling function which assigns labels to flows. The function* $P_g: S \times L \rightarrow [0,1]$ *is a partial function which for a node* $g \in G^D$ *and label* $l \in L$ *assigns probabilities to all outgoing sequence flows* $((g, x, l))$, *such that for a given l:* $\sum_{\forall x \in out(g)} P_s((g, x)l)) = 1$.

The definition of a *BPD* given in Definition1 models workflow processes by using elements of *F* to define a directed graph with nodes which are elements of *N*. However, Definition1 allows for graphs which are unconnected, do not have start or end elements, and are free-form or have various other properties which place them outside what is implied to be permitted in standard BPMN models. To ensure that a *BPD* describes a meaningful workflow process, we have developed a set of well-formedness rules (Herbert & Sharp, 2012b) which enforce restrictions on connecting elements, pool boundaries, and message passing.

The function *P* in Definition1 allows for the modelling of probabilistic decision points in the modelling of production and business processes. The intention is to capture real-world behaviour where the outcomes of complex decision within a process can appear random and are not possible to predict in advance. BPMN makes use of external conditions on decision gateways to select the outgoing flow from a decision point. These decisions are modelled by the set *L* and assigned to specific flows by the function *lab*. In practice, decision points

in a workflow process will have outcomes which depend on some inherent property of the task or on outside factors. The idea is that at a decision point an active choice is made, and then that choice results in a number of different possible outcomes. Figure 2 illustrates the application of P to a decision gateway g.
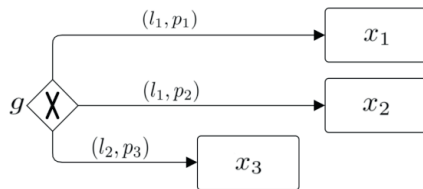

Fig. 2. Assignment of label probability pairs to a decision gateway. Here application of P requires $p_1 + p_2 = 1$ and $p_3 = 1$.

To enable quantitative analysis of a workflow process, we add numerical data to our models by using the following function which associates positive real numbers with tasks in a *BPD*.

Definition 2 (BPD Task Reward Function). For a *BPD* a reward function for a task $t \in \boldsymbol{T}$ is a partial function $R: \boldsymbol{T} \to \mathbb{R}_{\geq 0}$.

This function captures the notion that certain nodes have some reward or cost associated with the task. We may associate as many reward structures as we wish with a given *BPD*, so that a single task may have multiple different numerical properties which are incremented when the task is performed. Further details of these structures and model checking of these properties can be found in (Herbert & Sharp, 2012a).

3. Evolutionary algorithm

Our approach to the optimisation of production materials workflow processes is to mimic the process of natural evolution in the form of an evolutionary algorithm. An evolutionary algorithm is a subset of evolutionary computation; it's a generic population-based optimization algorithm. This approach has been used to address optimization in many different fields such as engineering, social sciences, robotics, biology, marketing and physics (Yo & Gen, 2010).

An evolutionary algorithm uses mechanisms inspired by biological evolution, which includes reproduction, mutation, recombination, and selection). Reproduction (or procreation) is the process by which new "offspring" individuals are produced from their "parents". Mutation is a change in the sequence of the process being investigated (e.g. organism, production or business process, code). Recombination is the process by which two processes exchange information, resulting in the production of a new combination of processes (e.g. DNA, tasks in a workflow process). Selection is the process by which traits become either more or less common in a population as a function of the effect of traits in relation to the desired goal (e.g. survival in biology, or increased production efficiency in a production materials workflow process). It is a key evolution mechanism.

Possible solutions, often called candidate solutions, for the optimisation problem the evolutionary algorithm is employed to solve, are viewed as individuals in a population and their suitability as a solution is determined using what is called a fitness function. A fitness function is a particular type of objective function that is used to summarise, as a single figure of merit, how close a given solution is to achieving the set aims (e.g. how close it is to fulfilling the optimization goals). The evolutionary aspect enters the picture because the above operators are applied multiple times. The evolutionary process is:

1. Generate the initial population of individuals randomly. This is called the first generation.
2. Evaluate the fitness of each individual in that population based on the optimization criteria given.
3. Repeat the fitness evaluation on this generation until it is terminated. Termination criteria can be time limit, sufficient fitness achieved, etc.
4. Select the best-fit individuals for reproduction; these are called the parents.
5. Breed new individuals through crossover and mutation operations to give birth to offspring from the selected parents. Crossover is an operator used to vary the programming from one generation to the next. Mutation is an operator used to maintain diversity from one generation of a population to the next.
6. Evaluate the individual fitness of the new individuals (the children).
7. Replace least-fit population with new individuals.

The evolutionary process is repeated until an individual is found which fulfils the fitness criteria within the given parameters. If for example we use the evolutionary process to optimise machine cycle time we would first define the optimisation goals. Hereafter we would define constraints and the parameters which the improved production materials workflow process, which the machine cycle is a part of, would need to obey. Different routes through the workflow process would then be tested against the optimisation goals until the process is stopped and the most optimal process is found which best fulfils the optimisation goals.

3.1. Optimisation using evolutionary algorithm

The evolutionary algorithm we have developed to optimise business workflow processes employs a genotype-style representation of the optimization problem, variation and selection operators, and a fitness function. However, many details are quite different from typical evolutionary approaches.

The algorithm we have developed performs optimisation of a BPD by performing modifications directly on the BPD ensuring that the final improved process is also a BPD and requires no special interpretation by end users. The description of the algorithm is intended only to explain the principle of this method and not detail the mathematic proofs behind it.

In the approach we have developed, an initial population of BPD variants is generated. Due to the computational expense

of performing quantitative model checking of a BPD we filter these variants using well-formedness (structural semantics) criteria and functional requirements, before evaluating their quantitative properties.

The evolution of the initial population takes place for a number of generations determined by limit. For each generation, a population of a given size is generated. This population is produced by selecting pairs from the previous generation, in a fashion that is proportional to their fitness score. This pair is used to generate a new variant BPD using a crossover operator. Mutation is achieved by performing a number of alterations of a BPD dictated by the mutation rates. If a variant proves to be well-formed and meets functional requirements it becomes part of the next generation.

Finally, once the generation limit has been reached, the highest scoring member of the final generation with regard to the optimisation goals, becomes the optimized BPD. Before returning this optimised BPD, any redundant components are removed.

To enable the combination of multiple weighted objectives, and to have sets of optimization goals in which both rewards and event probabilities can be expressed, we employ a set of optimisation goal tuples to define an individual optimisation goal using PCTL formulae. This is because in practice a production process is frequently optimised with regard to multiple quantitative properties. We use a set of optimisation goal tuples for this purpose. For a set of optimization goal tuples, we evaluate the relative improvement of a new production process BPD compared to an existing BPD using an optimisation goals scoring function.

Functional requirements allow the expression of properties which must hold for any future production process BPD derived from a BPD. Like optimization goals, functional requirements will be defined using PCTL formulae, however, in this case we will require that probabilities or reward values within the query are explicitly defined, such that the return value of the query is a Boolean variable. This is to ensure the functional requirements for each individual can be quickly evaluated as either being true or false.

A key step in our algorithm is the selection of members of a current generation used to derive the next generation. Here we employ stochastic sampling with limited replacement. In essence, each member of a current generation is mapped to a contiguous segment of a line, such that each individual's segment is proportional in size to its fitness. A random number is generated and an individual A whose segment spans the random number is selected. The process is repeated to obtain a partner with the restriction that if A is selected a new sample is chosen.

When generating variants we employ the traditional evolutionary algorithm approach of constructing a separate genotype representation upon which to perform modification of a BPD. Our approach allows the genome structure to closely reflect the phoneme structure. Encodings with this property are believed to make the evolutionary algorithm more robust (i.e. reduce the probability of fatal mutations), and also improve the capacity of a system for adaptive evolution.

We employ an adjacency matrix style representation of the underlying graph structure of the BPD for our genotype, where each matrix element is a vector which stores the reward structures associated with the given node of a BPD. The phenotype is simply the BPD that is derived from this matrix representation.

Crossover follows naturally from the structure of the genotype representation. Instead of creating a child by swapping information from two parents based upon one or more points in a linear structure as is commonly done, we use a rectangular section of the matrix structure selected at random. An offspring is then created by using information from inside the rectangle of one parent, and outside the rectangle of the other parent as illustrated in Figure 3.
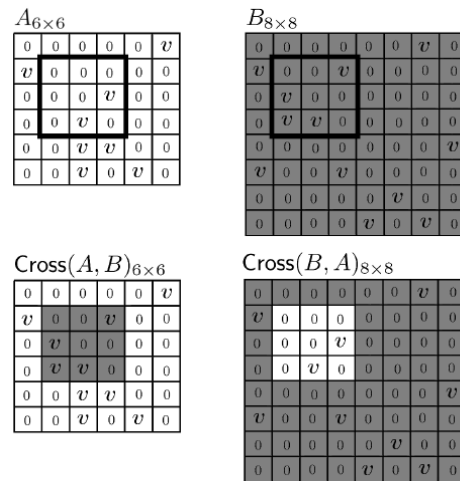

Fig. 3. Crossover.

Mutation is also defined as a mathematical operator, which is applied to specific elements of the matrix representation of a BPD. This compliments our crossover operator by injecting small local changes to a BPD. We define mutation to allow for considerable variation of a source BPD. This definition allows mutations to have two effects on a BPD:

1. Re-sequencing: Illustrated in Figure 4, this modification alters the BPD element which defines sequence flows. Specifically, it alters the relation between two nodes in the sequence Low (e.g. A and B in Figure 4(a)), replacing the destination node with a different node (Figure 4(b)), and reconnecting any excluded nodes to follow after the re-sequencing (Figure 4(c)). This has the effect of introducing a degree of randomness in the sequencing of a of tasks in the BPD.

2. Parallelization: This modification is illustrated in Figure 5 and functions by injecting pairs of parallel merge and fork gateways. These can be injected at any point other than at start and end elements (e.g. between A and D in Figure 5(a)), and the nodes between the injected gateways are initially all assigned to one of the parallel paths (e.g. Figure 5(b)). Note that when this is combined with the re-sequencing operator both parallel branches will eventually contain nodes.
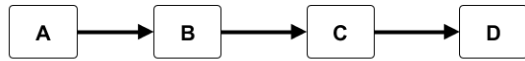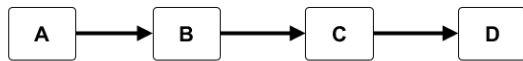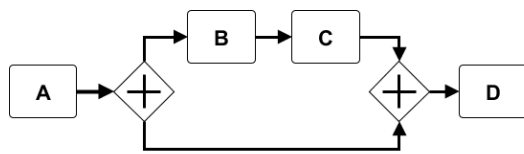

(A) Before
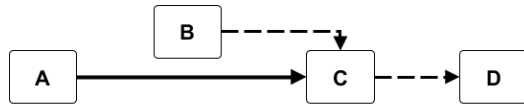
(A) During
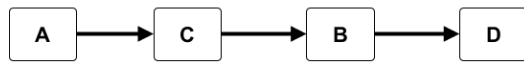
(A) After

Fig. 4. Re-sequencing.

(A) Before

(B) After

Fig. 5. Parallelization.

3.2. Implementation of the optimisation algorithm

In order to use our evolutionary algorithm to optimize production materials workflow processes in practice we have designed a prototype software tool called SBOAT which stands for Stochastic BPMN Analysis Tool. This tool allows practitioners to model, analyse and optimise business workflow processes. As the tool has a graphical GUI interface, the user does not need to have any prior knowledge of the technical workings of the tool but need only be able to associate rewards and probabilities to a workflow process in order to optimise it according to the desired parameters (Figure 6).
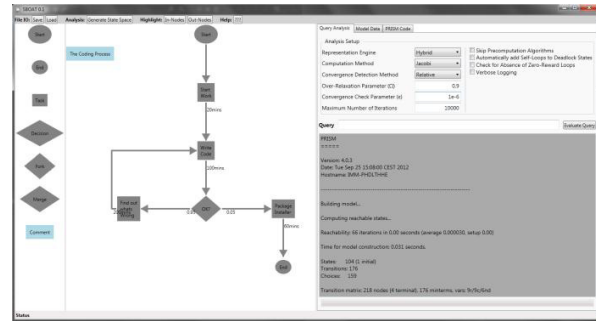

Fig. 6. User interface for the prototype version of SBOAT.

SBOAT is able to model and annotate with rewards and stochastic branching, a production or business process as a BPMN BPD. Analysis is specified using a PRISM style PCTL query and depending on the nature of the query one or a number of results are calculated. At the core of the tool is the PRISM model checker which performs analysis of individual models generated (Figure 7).
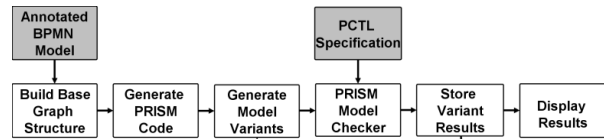

Fig. 7. Overall design

4. Case study

4.1. Description of industrial environment

The application of these methods was explored in a case study involving one of the largest Danish producers of baked goods which, for reasons of anonymity, is designated Baked Goods A/S. This company was established in 2000 and entered the market in 2001. In 2012, Baked Goods A/S had 103 full time employees. The company focused on increasing its domestic market revenue, and on developing new export markets, especially in China. Revenue increased to DKK 180 million in 2012. The increase mainly came from the domestic market despite the general downward price development on the Danish market. Revenue growth primarily derived from increasing sales of convenience products, but also from coffee bread products, including buttermilk horns, which proved successful in 2012. Baked Goods A/S sought to have a continued price focus and a high innovation level

The company is mainly focused on differentiation by making their bake-off products appear more "home-made" by making them less regular (e.g. not completely the same size or shape), thereby introducing a controlled amount of stochasticity into the production line. However, they also produce products with strict requirements for regularity for customers for whom this is vital. There are strict rules regarding listing nutrition facts on food in Denmark so it is difficult to differentiate products based on this. As a part of their differentiation strategy, Baked Goods focuses on

signaling the connection to home-made, fresh and natural products. This is done through labels on the products, promotional material and how the company presents itself and views itself (e.g. a company located in the country-side with a focus on traditional baking values and techniques). Baked Goods experiences very volatile commodity prices which makes them focus on new markets, different products, and higher efficiency. An example is that a sausage roll can only be sold by the firm for around 1-2 DKK each while shops take 6-8 DKK for this bake-off product, e.g. a mark-up of around 75%.

Baked Goods A/S have two production lines. Line 1 develops cakes and pastries, and line 2 develops baked goods like sausage rolls and pizzas.

4.2. Practical use of the optimisation algorithm

To illustrate an application of this method we will consider a specific example of a simple production process inspired from the baked goods case study involving bread production. To employ the method, one begins by building a BPD model of an existing production process. Figure 7 is an example of such a process which is annotated with rewards and information about its stochastic behaviour. This naively-designed production process consists of two processes, Conveyor Belt modelling the actions of the machines on a conveyor belt, and Filling Robot which models the actions of robot which fills the dough with cream etc. when needed.

For the production process described in Figure 8 it would be desirable to see an improvement in the time taken for a conveyer belt machine to complete baking a cake. This is in other words the optimisation goal. Further, it would also be desirable that the rate of filling consumption and the consequent probability, given a specific filling stock size, of running out of the filling is kept as low as possible. This is the second optimisation goal.

In addition to the optimisation goals, a number of functional requirements exist for this process (formally expressed using the temporal logic PCTL in SBOAT). These requirements describe the sequences the steps the process have to be in:

1. The baking of the dough should take place before the leaving process of the dough.

2. All dough making, leaving, cut and filling, must take place before a cake is packed:

3. The conveyer belt machine cannot pack the cake before it has received filling.

4. The Filling Robot must ensure that a filling has been prepared and measured before it is sent to the conveyor belt.

5. When the conveyer belt determines that a bad dose of filling has been received it must immediately request a new dose.

Note that the final functional requirement (item 5) is not currently satisfied by the initial BPD shown in Figure 8.

Figure 9 illustrates one possible outcome of applying our optimisation methods to the BPD shown in Figure 8. Specifically, this is the outcome of 28 generational improvements of population size 500 of the process. Note that the new functional requirement (item 5) is now satisfied. In the case of this example the rates for sequencing and parallelizing are set so that the Mutate function ensures that considerably more re-sequencing modifications are performed than parallelization modifications.

In this run of our optimisation method we have identified two opportunities to parallelize actions. Within the Filling Robot process, the filling can be prepared and measured at the same time. In the conveyer belt process, it is possible to prepare dough and cut it while the dough simultaneously leaves. In both cases this saves time, as when performing actions in parallel only the path with the slowest behaviour is counted towards the parallel sections contribution to the reward value.

We have also determined that, in this simplified example, the conveyer belt process always orders filling to finish the cakes but the conveyer belt machines must wait while the Filling Robot performs its operations and then returns the filling. As the filling will inevitably be needed, and only the packing of the cake needs to be done after the dough has been made, it is within the functional requirements for the process, and results in a considerable time saving to order the filling immediately before even preparing the dough. This ensures that there will be no delay imposed on the Conveyer Belt process by the actions of the Filling Robot process. This simplified optimisation example does not violate the functional requirements and results in a significant reduction of the time taken for the execution of the production process.

This example highlights some of the strengths and weaknesses of this optimisation method. Existing languages for the modelling of business processes such as BPMN, UML activity diagrams or YAWL lack a formalised semantic basis which would enable formal analysis and subsequent automated scheduling. Further, these languages do not allow for modelling stochastic behaviour or provide mechanisms to effectively track the consumption of resources during execution. These aspects are therefore the key strengths of this optimisation method as no other method, to our knowledge, has all these features. Further, it should be noted that our method by employing the PRISM tool calculates exact values. However, this need for precision also means that a disadvantage of our approach is that it requires detailed knowledge of the workflow processes being optimised. Another disadvantage of our method is that to use the optimisation schedule in practice great computing power is needed which can be both expensive and time-consuming. However, our method allows for automatic optimal scheduling with mathematical precision and within specific parameters which can help organisations limit waste of, for example, energy or material as well as optimise production with regard to parameters such as time, human resources and cost.
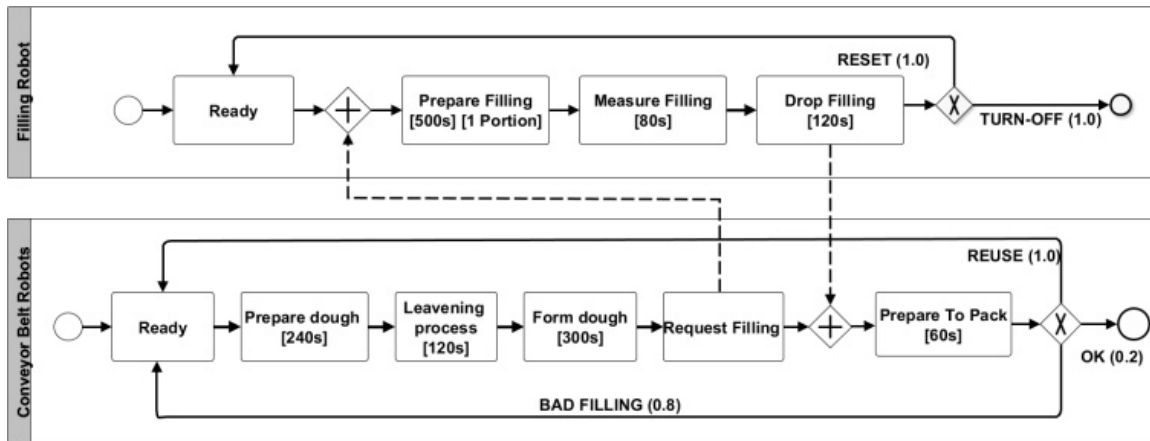
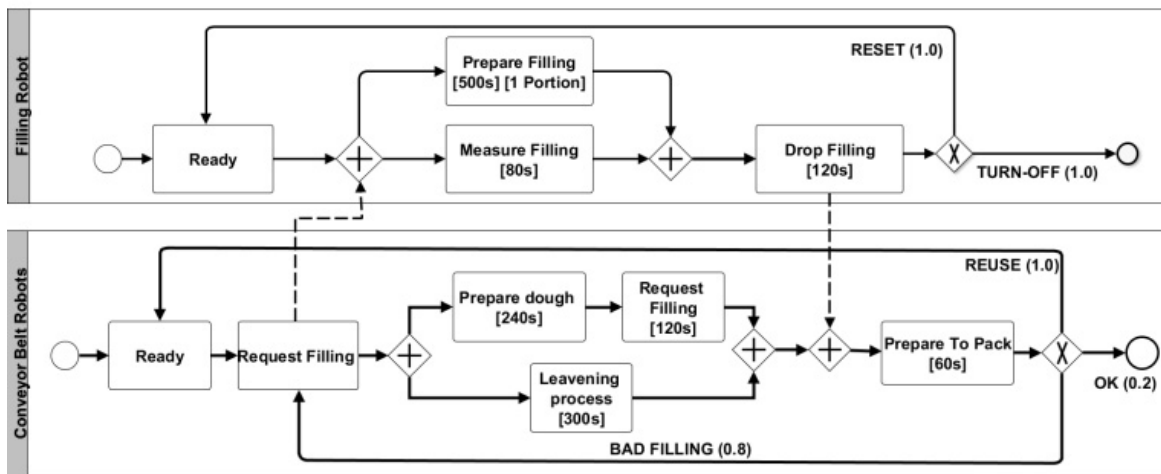Fig. 8. BPD model of a baking workflow process before optimization.

Fig. 9. BPD model of a baking system after 28 generations of population size 500 optimisations.

5. Conclusion and notes for further research

In this paper we have outlined a framework for the automatic optimisation of production materials workflow processes through an evolutionary algorithm. We have based the presentation of our optimisation method on the language BPMN.

Our work provides a mathematical foundation to suggest that Gilbreth's (Gilbreth, 1921) vision of finding the one best way to do work, e.g. an ideal work process, is similar to searching for an ideal organism in a given biological ecosystem. However, there will, in any sufficiently complex market, be multiple ideal workflow process structures depending on which parameters the organisation wishes to optimise.

The case study has shown the strengths with our optimisation method. Compared to other approaches for production optimization our approach allows for the formal mathematical analysis and subsequent automated optimization while allowing for modelling stochastic behavior as well as providing mechanisms to effectively track the consumption of resources during execution.

Further research will focus on refining this optimisation approach and SBOAT. We hope to release this tool in late 2014 and use it in several case studies and develop a set of benchmark BPMN optimization problems. This will allow for a more extensive formalised exploration of the scope and parameters of this method.

Acknowledgements

References

[1] van der Aalst, WMP, van Dongen, BF, Herbst, J, Maruster, L, Schimm, G. & Weijters, A. (2003), Workflow mining: a survey of issues and approaches, Data & knowledge engineering 47 (2), 237-267.

[2] Aziz, A. Singhal, V. Balarin, F. Brayton, R. Sangiovanni-Vincentelli, A. (1995) It usually works: The temporal logic of stochastic systems. In P. Wolper, (ed.), 7th International Conference on Computer Aided Verication (CAV'95), 939: 155-165. Springer.

[3] Ball T. and Jones, R. (2006). Eds., ser. Lecture Notes in Computer Science, London, UK: Springer-Verlag, 234-248.

[4] Basu, A., Bensalem, S., Peled, D. & Sifakis, J. (2011). Priority scheduling of distributed systems based on model checking. Formal Methods in System Design 39(3): 229-245

[5] Baier C. and Katoen, J.-P. (2008). Principles of Model Checking. Cambridge MA, USA: The MIT Press.

[6] Christensen, MM, Haar, S. (2013) Supermarkeder presser brødkæmpe i minus. Børsen, 8/08/2013. 151:12-13.

[7] Christiansen, DR, Carbone, M. & Hildebrandt, T. (2011), Formal semantics and implementation of BPMN 2.0 inclusive gateways, in Proc. of the 7th international conference on Web services and formal methods, ser. Web Services and Formal Methods 2010, Berlin, Heidelberg: Springer-Verlag, 146-160.

[8] Gilbreth, FB, Gilbreth, LM (1921) Process Charts: First steps in finding the one best way to do work. American Society of Mechanical Engineers (ASME).

[9] Gröÿer, M., Norman, G., Baier, C. Ciesinski, F. , Kwiatkowska, M. & Parker, D. (2006), On reduction criteria for probabilistic reward models, in Proceedings of the 26th international conference on Foundations of Software Technology and Theoretical Computer Science, ser. FSTTCS'06, Berlin, Heidelberg: Springer-Verlag, pp. 309-320.

[10] Hansson H. and Jonsson, B. (1994), A logic for reasoning about time and reliability, Formal Aspects of Computing, 6(5): 512-535.

[11] Herbert, L & Sharp, R (2013a) Optimisation of BPMN business models via model checking. Proc. for ASME 2013 International Design Engineering Technical Conferences and computers and information in Engineering Conference (IDETC/CIE2013)

[12] Herbert, L, Sharp, R (2013b) Precise analysis of probalistic business process model and notation workflows. Journal of Computing and Information Science in Engineering, 13: 2-11

[13] Herbert, L. & Sharp, R (2012a) Quantitative analysis of probabilistic (BPMN) workflows. Proc. for ASME 2012 International Design Engineering Technical Conferences and computers and information in Engineering Conference (IDETC/CIE2012).

[14] Herbert, L & Sharp, R (2012b), Using stochastic model checking to provision complex business services, in High-Assurance Systems Engineering (HASE), IEEE 14th International Symposium, 98-105.

[15] Kwiatkowska, M. Norman, G. and Parker, D. (2006). Symmetry reduction for probabilistic model checking, in Proc. 18th International Conference on Computer Aided Verification (CAV'06).

[16] Kwiatkowska, M., Norman, G. & Parker, D. (2011). PRISM 4.0: Verification of Probabilistic Real-time Systems. In Proc. 23rd International Conference on Computer Aided Verification (CAV'11), vol. 6806 of LNCS, 585-591, Springer.

[17] Kwiatkowska M. & Parker, D. (2012), Advances in probabilistic model checking', in Software Safety and Security - Tools for Analysis and Verification, T. Nipkow, O. Grumberg and B. Hauptmann, Eds., ser. NATO Science for Peace and Security Series - D: Information and Communication Security, IOS Press, vol. 33: 126-151.

[18] Muehlen, M. & Recker, J. (2008). How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation. Proceedings for CAiSE '08 Proceedings of the 20th international conference on Advanced Information Systems Engineering, 465-479.

[19] Object Management Group (2011) Business process model and notation (BPMN) 2.0. Standards Document formal/2011-01-03. Webpage: http://www.omg.org/spec/BPMN/2.0/

[20] Parker, D. (2012). PRISM 4.0 manual, available: http://www.prismmodelchecker.org/mamual

[21] Shingo, S. (1981) Study of Toyota Production System from Industrial Engineering Viewpoint. Japan Management Association.

[22] White D.J. (1993), Markov decision processes. John Wiley & Sons.

[23] Wijs, A.J., Pol, J.C. van de & Bortnik, E.M. (2009). Solving scheduling problems by untimed model checking : the clinical chemical analyser case study. International Journal on Software Tools for Technology Transfer, 11(5), 375-392.

[24] Yu, Xinjie, Gen, Mitsuo (2010), Introduction to Evolutionary Algorithms, Decision Engineering Series, Springer.