Derivation of probability-based configurations for early feasibility statements in the development of products with high variance

Niels Demke, **Florian Tichla**, **Jörn Zühlke**, **Thorsten Schmidt** and **Frank Mantwill**
Professorship for Machine Elements and Computer Aided Product Design,
Helmut Schmidt University / University of the Federal Armed Forces Hamburg, Germany
{niels.demke, florian.tichla, joern.zuehlke, thorsten.schmidt, frank.mantwill}@hsu-hh.de

Abstract

The automotive industry is characterized by variant-rich products and a development process with a high degree of complexity. Limited information related to the variant leads to suboptimal decision making during early product development stage. Therefore, an existing potential to meet customer wishes more efficiently is not utilized. In order to cope with this situation, this paper presents an algorithm which uses data on product features and their installation rates as well as rules and sales figures of physical products. This results in a set of probable variant configurations at a very early stage in the development process. These configurations can then be used for variant planning and feasibility assessments. The method is implemented within a Matlab application. Using logic based knowledge processing as well as feature frequencies, a SAT-based approach verifies whether the result is within a feasible solution space.

1 Introduction

The ability to create a large number of variants of one's products is a major competitive advantage on saturated markets, so that products with a high variant count are offered by the automotive industry [Stich, 2007]. For example, an automotive has approximate 10^9 product variants [Zagel, 2006]. Individual products can be described with an open variant configuration in the product documentation via their features [Herlyn, 1990]. The product documentation includes a Product Layer which contains the features and restrictions for the configuration, the Technical Layer with the configurable bill of materials (BOM) and the Geometry Layer with CAD-Models and Drawings (Figure 1).

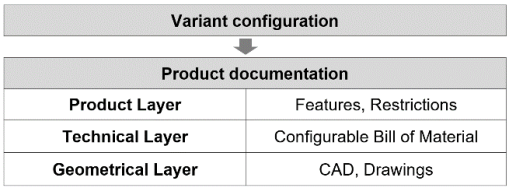

Figure 1: Variant configuration according to [Herlyn, 1990]

The properties defined at the beginning of the product life cycle determine the quality, cost and future market success of products. The ability to influence the product is reduced with increasing maturity, because the costs of change increase steadily as described in the Rule of Ten [Ehrlenspiel *et al.*, 2014]. Accordingly, most variants are to be generated at the beginning of the product development process in order to ensure an optimal ratio between costs and benefits regarding variety. The variant management distinguishes the theoretical variance, the technical feasible variance and the variance that is offered to the customer. This paper addresses the challenges of dealing with complexity in handling of theoretical variance and technically feasible variance. In the planning process, the frequency with which each feature is chosen, must be estimated. The estimate is based on experience and initial assumptions about the future product. The variant management includes strategies and methods for dealing with variance at an early stage. These are the avoidance, reduction and control of variants (Figure 2) [Eisenhart, 2002].

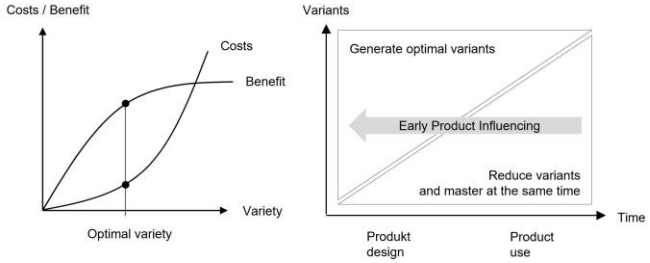

Figure 2: Objectives of variant management according to [Heina, 1999; Herrmann and Seilheimer, 2002]

Variant management supports the processes between the specification of product requirements and the development of technical restrictions at the product documentation levels. This process is iterative in the early phase of the design process, which requires specific methodical support. Therefore, the focus of this paper is to transfer the practical problem of reflecting technical relations between different layers of product documentation. In order to achieve this, the paper describes a method to create a large set of feasible variable configurations within the product layer.

This process poses two problems which have to be solved. First, the estimated feature frequencies must not contradict the ruleset which incorporates the technical conformity. Second, the ruleset as well as the estimates are subject to change and therefore require continuous re-analyses. Insufficient analyses result in an information deficit and have an impact on subsequent steps such as parts requirements calculations, financial evaluations, purchasing planning and product planning, because they simply cannot be performed accurately without sufficient information. Consequently, the potential to influence the product early in the process is not fully exploited. This gives rise to three significant research questions:

1. How can logical formulations be used in order to model the constraints in variant management?
2. Which methodical requirements result from feasibility analyses of variant-rich products, the processing of probability-based statements and the rules for variant management?
3. Which property of a new set of created variable configurations can be tested in order to determine the quality of the found solution?

2 Related Work

The approach of this paper is to address different disciplines of engineering as well as computer science. By combining these disciplines, the topic is treated comprehensively and a need for further research is subsequently highlighted.

2.1 Open variant configuration

In order to approach the topic of variant-rich products, it is first necessary to clarify the use of the term "variant". Herlyn [1990] defines variants at each level of production, from the raw part to the finished product. This means that there are different types of variants. The variants of products are only one type of variants. The Cambridge Dictionary definition of a variant is: "something that is a slightly different form of another form" [Cambridge Dictionary, 2021]. There is always at least one counterpart of which something is a variant. They have common properties but are also mutually exclusive by some form of deviation. Together with the knowledge that there are variants in all production levels of the product description, it can be stated first that "variants of products" are end products of a production, which are similar in a certain portion, but always have differences from one another.
The representation of products in BOMs is a part of the technical information. Since there is a high degree of overlap in the information to be documented for product variants, there is an unjustified effort in generating and maintaining individual BOMs for each product variant [Eisenhart, 2002].
The literature classifies different types of BOMs. These include type and multiple BOMs, common parts/variant BOMs (GT/VT BOM), and selection or complex BOMs (cf. [Eisenhart, 2002; Herlyn, 1990; Stich, 2007]). The complex BOM is widely used, for series products with a large number of variants (e.g. in the automotive industry). It is characterized by the fact that all product variants are managed in one BOM. They are defined "in the complex" [Herlyn, 1990]. The description of the product variants is done outside the BOM in the so-called "feature level" [Eisenhart, 2002]. In this context, Herlyn [1990] also calls the BOM the "technical level". It refers to the features of the product variants, which must first be recorded (Table 1).

Feature	Value	Description
TRW		Partition wall
	3CA	without partition
	3CX	Mesh partition
ATA		Drive type
	1X0	front-wheel drive
	1X1	All-wheel drive
KAR		Body shape
	K8B	Notchback
	K8D	Golf Variant
	K8G	Hatchback
	K8M	Bora Variant

Table 1: Features and values according to [Eisenhart, 2002]

The restrictions between the feature values are also documented in the feature level. The restrictions limit the possibility to combine feature values of different features (Table 2). In the restrictions "+" means logical "and" and "/" means logical "or".

Restriction	Description	Boolean notation
K8B Z 3CA	Notchback forces without partition	K8B → 3CA
1X1+K8D Z 7B2	All-wheel drive and Golf Variant forces 12 volt socket in trunk	1X1∧K8D →7B2
K8G Z 3CA	hatchback forces without cargo floor	K8G → 3CA
K8M Z 7B2	Bora Variant forces 12 volt socket in trunk	K8M → 7B2
3GN Z K8D/K8M	Variable load floor concept forces Golf Variant or Bora Variant	3GN → K8D∨K8M
3GN Z 1X0	Variable load floor concept forces front-wheel drive	3GN → 1X0

Table 2: Restrictions according to [Eisenhart, 2002]

The example of Eisenhart [2002] represents a section of the feature level. An example of the link between the part numbers at the point of use "load compartment trim" and the characteristic values can also be found in Table 3. The column "Part validity" assigns the feature values from the feature level to the part numbers in the form of Boolean functions. Similar examples of this systematics in the complex BOM can also be found in [Herlyn, 1990] and [Stich, 2007]. Stich [2007] speaks of „code rules“ in this context.

Part number	Description	Part validity	Quantity
1J0.343.132.A	load compartment trim	3CA+1X0+3GA+7B2+ K8D/K8M	1
1J0.343.132.B	load compartment trim	3CA+1X0+3GN+7B2+ K8D/K8M	1
1J0.343.132.C	load compartment trim	3CA+1X1+3GA+7B2+ K8D/K8M	1
1J0.343.132.D	load compartment trim	3CX+1X0+3GA+7B2+ K8D/K8M	1
1J0.343.132.E	load compartment trim	3CX+1X0+3GN+7B2+ K8D/K8M	1

Table 3: Complex BOM according to [Eisenhart, 2002]

Building orders are described in the open variant configuration via a concatenation of feature values. If a customer wants to define their final product, configurators based on the restrictions lead them to a buildable configuration. For each customer order, the part validities are evaluated in the complex BOM. They represent Boolean functions that assign the values "true" or "false" based on the features of the build order. If a part is required for a build order, the function returns the value "true" otherwise "false". In this way, the material consumption can be calculated for each valid construction order.
Frischen *et al*. [2019] describes the handling of rule-based complex BOMs in the early phase of product development. For this purpose, mechanisms are shown that affect both the product layer and the technical layer. They all have the goal to integrate the documentation in the form of rule-based complex BOMs along the entire product development process. In this context, cost-benefit assessments, product planning processes, and forward sourcing must be applied as early as possible in the product development process (cf. [Ehrlenspiel *et al*., 2014]). For such considerations, the installation rates of the features estimated by the sales department are an important element of the open variant configuration. Multiplied by the expected total number of units, they can be used to estimate the subsequent frequency of feature call-offs. The material requirements derived from them are also relevant for the calculations mentioned above. For cost-benefit calculations, the material costs can be estimated. For forward sourcing planning, the material requirements can be better estimated over the lifetime of a product.

2.2 Planning approaches

In production, planned orders are used to simulate uncertainties in the order situation. In the automotive industry, simulation of production orders is used for sales planning, quantity planning, capacity balancing and material requirements forecasting. Installation rates are used to describe the variance in the expected customer configurations [Bayer *et al.*, 2003].
Wagenitz [2007] describes a modeling approach to order processing by first describing the sales order processing and proposing planned orders to estimate part requirements by filtering the BOM for a given configuration. The used installation rates are derived from historical vehicle orders. The findings are used for distribution and material planning from a logistics perspective.
Bürgin [2018] uses the scheduling of planned orders for scenario generation for customer orders and thus for planning the uncertainty between short-term and medium-term order fluctuations (as part of the order planning). Bürgin *et al.* [2017] takes feasibility restrictions regarding the combination of features as well as probabilities when choosing feature values into account. Furthermore, according to Bürgin, it is possible to embed the planned orders in a production network, whereby incoming sales orders can be assigned to preproduced planned orders. Through the production networks, a gain in flexibility can be achieved through forecast production orders as "build-to-stock", in particular through cross-location program planning [Wittek, 2013].
Kappler *et al*. [2010] describes a robust planning of primary and secondary demand planning by an integration of the product documentation. For this purpose, Kappler chose an interval-based approach to production program planning and derives demand barriers for components from this. He describes a "build-to-forecast" principle for suppliers according to a range of installation rates.
Voronov [2013] proposes different formal methods for large-scale product configuration. In addition to the problems of verification in configurations, he introduces Constraint Satisfaction in search based Boolean Satisfiability Solvers as a possibility of automatic reasoning about interactive product configuration.
Küchlin [2020] employs the advanced use of conventional symbolic Artificial Intelligence (AI) in the form of propositional logic and automatic reasoning through SAT solving for the use in analysis and verification in the automotive industry. See also Automated Configuration Problem Solving (cf. [Petrie, 2012]).
An examination of the related work reveals that the authors predominantly consider the aspects of production. Thereby, a feedback of the findings into the development process and thus the proactive product influence in the early phase of product planning is not considered sufficiently.

2.3 Logic-based approaches

For logic-based description and solution of general problems describable by constraints, a number of frameworks are used. The most general approach is the Constraint Satisfaction Problem (CSP) which is a more generalized form of the Boolean Satisfiability Problem (SAT) [Astesana *et al.*, 2010].
The Boolean Satisfiability Problem is a fundamental, NP-complete problem from the core area of information technology. Finding a solution to this problem lies in in a valid variable assignment of a propositional logical function. It is possible to solve this problem in a finite amount of time using the DPLL algorithm [Davis *et al.*, 1962]. Based on this algorithm, there have been a continuous number of extensions and improvements in search of valid solution methods: The Conflict Driven Clause Learning (CDCL) [Marques-Silva and Sakallah, 2003], the Boolean Constraint Propagation (BCP) [Moskewicz *et al.*, 2001] and for a comprehensive insight [Biere *et al.*, 2009; Chen *et al.*, 2001; Eén and Sörensson, 2004; Große *et al*., 2011].
In practice, SAT solvers are commonly used to check the feasibility of configurations where the number of configurable variants is too large to be described individually and which are therefore formulated with a set of constraints (see section 2.1). To make SAT solvers more accessible from the frontend for e.g. the use of probabilistic statements, SMT solvers exist as an extension. In this way, more complex scopes can be solved against the background of "modulo theories" using the solution methods of the Boolean Satisfiability Problem [Barrett and Tinelli, 2018]. The advantages of the established solvers on the one hand are efficient and fast processing of conjunctive normal formats (CNF) and deterministic statements about valid solutions with supplied proof as one possible solution. On the other hand, the available product data

from the industry require a complex conversion into the CNF format of the solvers, which is a problem with the large size of the data set. To adapt the solution and subsequent use in following parts of the program, a separate ad hoc algorithm is presented, which meets the complex notation of the product encryption (cf. [Sinz *et al*., 2003]).

To date, no general solution for the described problems has been determined. To close this gap, a new approach has to deal with the variant management of vehicle properties from the product documentation and with the subsequent verification using a logic-based solver for validation. Previous work has not addressed this deficiency.

In this paper, the intention is to suggest a means to deal with installation rate estimates and open variant configurations in the variant management. The contribution of this paper is to include the installation rates in the generation of feasible build orders. Based on the further use of feasible build orders for follow-up processes and the development and encoding of product properties, a set of build orders can be derived.

3 Proof of Concept

This section introduces the necessary equations as well as the algorithm which is modelled to create the desired variable combinations. Firstly, a set of sales orders is to be generated in a way that none of them contradict any restrictions. At the same time, the frequency of the feature values contained therein should correspond to the specified installation rates. Thus a set of customer orders would be available, which represent a detailed estimate of the later customer orders early on in the product development process. Therefore, the complex BOM could be simulated for each of these orders. The result of these calculations could then support the evaluation processes early on in the development process. Specifically, a branching tree of Boolean rules is combined with a set of probabilities in order to predict future sales orders. Both concepts are combined in the application mentioned below. In order to be able to customize the application further down the line, it was determined that it was more efficient to create a custom algorithm rather than to adapt an existing solver. This means that the algorithm could be tailored to specifically solve the given problem. The input for the algorithm is as follows: There are n_v variables (Formula 1). They are either chosen (1) or not chosen (0) to be part of any given result.

$$x_i \in \mathbb{B}, \qquad i = 1 \dots n_v \tag{1}$$

Additionally, there are two kinds of rules that determine which variables are allowed to be part of the same result. First, there are *m* groups of variables called families (see features and feature values in Table 1). A family describes a group of variables which are mutually exclusive. The solution must include a variable from every family in order to be feasible (Formula 2).

$$\bigcup_{j=1}^{m} F_j = 1 \tag{2}$$

Additionally, the added probability for all variables of a family equals 1 (Formula 3). Every variable belongs to exactly one family. Within the algorithm, this relationship is specified within the matrix $\underline{FP}$.

$$\sum_{i=1}^{n} P(x_{i,j}) = 1 \quad , x_{i,j} \in F_j, for\ every\ j \tag{3}$$

Second, there are r rules. They represent the internal knowledge created out of restrictions for customer configurations mentioned above (see Table 2). They prohibit specific combinations of variables to be part of the same configuration. Additionally, they can be combined into a *CNF* (Formula 4):

$$\bigwedge_{k=1}^{r} \bigvee_{i=1}^{n} (\neg) x_{ki} \tag{4}$$

Overall the method is looking for a specific set of variables that returns "true" for the *CNF*-Representation. Within the algorithm below, a Boolean Matrix $\underline{R}$ is used to represent the *CNF*. It connects the different variables and determines which ones cannot be part of a feasible solution at the same time. The objective of Algorithm 1 is therefore to create a given number of solutions.

Algorithm 1: Configuration Builder

Input: Ruleset R, family-feature-matrix FP,
feature probability p(f)
Parameter number of required results n
Output: n feasible feature sets $\{f_1, \dots, f_{max}\}$ S
1: Initialize algorithm
2: **Outer Loop**: *Goal: Create n feasible solutions*
3: Initiate S_i *for i = 1 to n*
4: **Inner Loop**: *Goal: Create 1 feasible solution*
5: Find next variable to be part of solution S_i
6: Perform Algorithm 2: Update $\underline{FP}_{mod}$ using $\underline{R}$ and S_i
7: **end Inner Loop**
8: Add S_i to S
9: **end Outer Loop**
10: **return** solution S

To summarize, the input of the algorithm includes a matrix $\underline{R}$ which determines which variable combinations are not allowed to be part of the same solution. Additionally, the matrix $\underline{FP}$ determines which variables are part of each family and table *p(f)* determines the probabilities for each variable to be chosen. Line 1 includes all preliminary actions that are necessary for the implementation to run. Line 2 initializes the algorithm itself. It determines how many solutions have to be found and where these solutions are going to be stored. Line 3 sets up the local storage for the single solution that is created by the inner loop. Line 4 is the starting point for inner loop. Line 5 determines the next variable which will be part of the found solution. This process includes multiple steps. Firstly, in order to make sure that the solution remains feasible, it is necessary to check if there are families which only include one variable (see below at Algorithm 2 how variables are removed from $\underline{FP}$). These are then added to the solution. If all remaining families have more than one feasible variable, one of the families is chosen biasedly. The probability for each family to be chosen is reciprocal to the size of said

family. Once a family is chosen, the feature probability *p(f)* is used to randomly choose a variable within the family. Afterwards, the available variables have to be updated. This happens in Line 6 which starts Algorithm 2. Line 7 checks if the local solution is completed and loops if it is not. Line 8 adds the newly found complete solution to the collection of solutions. Line 9 checks if enough solutions have been generated and loops accordingly if the number of found solutions is smaller than the target number. Algorithm 2 determines which variables can still be chosen to be part of the solution S_i.

Algorithm 2: Update $\underline{FP}_{mod}$ using $\underline{R}$ and S_i

Input: modified family-feature-matrix $\underline{FP}_{mod}$, Ruleset $\underline{R}$
partial Solution S_i
Output: updated family-feature-matrix $\underline{FP}_{mod}$

```
1:  for each rule r in R do
2:      if all but one variable v of r are part of S_i then
3:          remove v from FP_mod
4:      end if
5:  end for each
6:  return updated FP_mod
```

Line 1 checks every row of $\underline{R}$. Every row of $\underline{R}$ defines a specific combination of variables that cannot be included in a solution. Therefore, Line 2 checks if all but one variable that are part of said row are also part of the solution. If this is the case, Line 3 removes the according variable from the modified family-feature-matrix $\underline{FP}_{mod}$. It is returned to Algorithm 1 in Line 6.
Overall, the interaction between both algorithms ensures that the results are always feasible. To determine if the algorithm is able to create reliable solutions, the relative frequency for each variable has to be close to the given probabilities derived from the use case.

4 Experimental Validation & Discussion

In order to determine if the model can reliably produce useful variable configurations, two properties of the resulting sets have to be examined. First, it has to generate feasible results i.e. follow all rules set in $\underline{R}$. This is a mandatory requirement. Due to the small subset of feasible results compared to the all possible combinations (estimated 10^{62} vs. quoted 10^9 [Zagel, 2006], (cf. [Kübler *et al.*, 2010])), it is very unlikely to generate feasible solution if they are created without a specific generator. Therefore, Algorithm 2 makes sure that only feasible solutions are generated in the first place. This requirement is therefore already fulfilled.
Second, when generating large sets of results, the frequency for each variable has to be close to its probability as they are set within *p(f)*. This is important if the results are supposed to be used as a basis to predict future sales or material needs. Therefore, in order to be actually valuable as a method to model future sets of variables, the rate with which any variable is chosen, has to converge onto its probability when a large number of results is generated. In order to determine if the model fulfils the second requirement, several steps had to be made. From a dataset of 175 families and 571 feature values from the automotive industry, a large number of result were generated. The used installation rates are based on 251,183 customer orders. After a new result was generated, the relative frequency (*RF*) of each variable is calculated and saved. The results are plotted after 20,000 cycles for a selection of four variables in the dataset. Figures 3 and 5 show this behavior. After approx. 20,000 configurations are created, the calculated installation rates show a variance of .0024 and a standard deviation of .0487 compared to the ones used to describe the model, which confirms the presumed convergence shown in Figures 4 and 6.

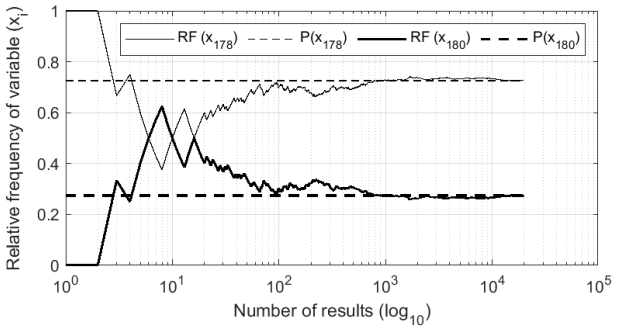

Figure 3: Good fit of the observed frequency of x_{178} and x_{180}

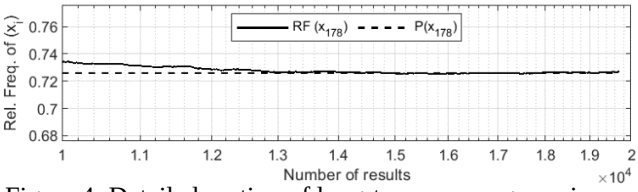

Figure 4: Detailed section of long-term convergence in x_{178}

The expected behaviour can be seen for every set of variables. However, some groups of variables show small deviations from the expected convergence towards its probability.

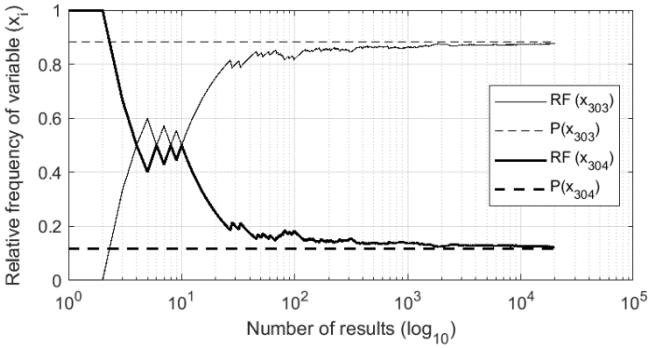

Figure 5: Poor fit of the observed frequency of x_{303} and x_{304}

Figure 6 shows a detailed view of the final points of data for one family of variables. One can see that there is a small but stable deviation from the given probability. The variables seem to converge onto a different value than the given probabilities for x_{303} and x_{304}. Since the general applicability of the approach has not been tested, it will be part of future research.

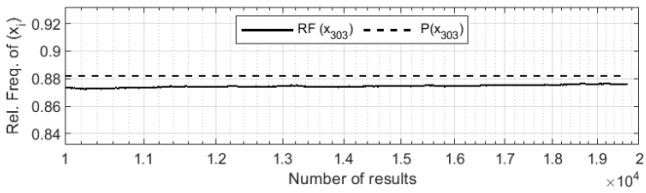

Figure 6: Detailed section of long-term deviation in x_{303}

There are multiple possible explanations for this phenomenon. First, it is quite possible that restrictions prohibit an outcome closer to the expected results. This would indicate systemic errors within the source material. Second, an alternative explanation could stem from the fact that the original data had a ruleset which changed over time as well as changing probabilities while both are presumed to be static. Both possible answers warrant further research.

5 Conclusion & Future Research

Research question 1 dealt with the logical formulation of constraints in the variant management. Therefore, the open variant configuration, planning approaches and logic-based SAT-Solver are analyzed. The analysis suggests the possibilities of describing early feasibility statements of highly variant products by a transfer description as a Constraint Satisfaction Problem, for example by SAT or SMT. Subsequently both advantages and disadvantages of established solvers were identified.

For answering research question 2, the adaptability of the approach was elaborated as a methodological requirement. The use of this approach in variant management also allows for better integration of collaborating organizations into the development process by making product data available in a meaningful format. The existing constraints on the use of product data in the automotive industry lead to limitations in the continuous use of existing SAT approaches.

Therefore, an ad hoc approach is presented, which on the one hand meets the requirements of the complex notation in the product encoding and on the other hand offers the possible flexibility to be adapted and used in further program parts in order to feed back into the design process.

Convergence between the results of the approach and the initially used installation rate was satisfactory, and found to be sufficient to satisfy the property of a new set of created variable configurations, as posed by research question 3. The necessary condition of buildability is ensured and fulfilled by the set of rules used in the ad hoc algorithm.

The unanswered aspects of question 1 to 3 are part of future research. This includes investigation into the use of SMT solvers such as the CVC4 to modulate probabilities and make validation and analysis of build orders even more efficient and fast. The estimation of the used installation rates was not part of this paper and shall be investigated in more detail by a modeling approach. In addition to optimizing the results, the approach will be extended to use dependent functions instead of constant build rates. For future studies in this area, there is an interest in extending the approach from constant to variable installation rates, which are a function of additional independent variables.

In comparison to the conventional algorithm used previously, this approach has the potential for the addition of extension modules to the procedurally interpretable code. From the methods of AI, such a method exists for the prediction of probabilities of a discrete binary available expression depending on one or more independent variables, for example, via the consideration of a logistic regression (LOGIT model). Logistic regression methods are already successfully used for classification (cf. [Mohri *et al.*, 2001; Hude, 2020]). The prospect of replacing constant installation rates by a regression function offers the possibility to introduce extension modules, to examine their suitability as regressands for the explanation of an installation rate by means of regression coefficients and to include them in the model for improvement in the sense of a learning effect.

This approach represents a set of common premises for feasibility analyses. Their simultaneous use in all business sectors involved within the product development process is the origin of the considerations presented in this paper and is to be the subject of further consideration. Likewise, the application of the gained knowledge within the product development process shall be the subject of future research.

The goal of this approach is a contribution in an increasingly hybrid decision-making process at the interface of humans and technology via the inclusion of a digital assistance, which is of great importance for the automation potential via integration of AI in the future product development process. From this outlook, a future need for research is justified in the interdisciplinary use of AI in product design, especially in an early phase of product planning and variant determination through simulation and classification approaches by generating and analyzing planned orders.

References

[Astesana *et al*., 2010] Jean Marc Astesana, Laurent Cosserat, Hélène Fargier. Constraint-based vehicle configuration: a case study, 2010.

[Barrett and Tinelli, 2018] Clark Barrett, Cesare Tinelli. Satisfiability Modulo Theories, 2018.

[Bayer *et al*., 2003] Johann Bayer, Thomas Collisi, Sigrid Wenzel. Simulation in der Automobilproduktion. Springer, Berlin, Heidelberg, 2003.

[Biere *et al*., 2009] A. Biere, M. Heule, H. Van Maaren, T. Walsh. Handbook of Satisfiability. 185 of Frontiers in Artificial Intelligence and Applications. IOS Press, Amsterdam, 2009.

[Bürgin *et al.*, 2017] Jens Bürgin, Han Hao, Gisela Lanza. Integrated Order Planning for Multi-Variant Products – An Approach for Generation of Planned Orders, their Allocation in a Production Network and their Matching with Customer Orders. [Integrierte Auftragsplanung variantenreicher Produkte]. In Industrie 4.0 Management 33, 2017.

[Bürgin, 2018] Jens Bürgin. Robuste Auftragsplanung in Produktionsnetzwerken. Mittelfristige Planung der varianten-

reichen Serienproduktion unter Unsicherheit der Kundenauftragskonfigurationen. Shaker Verlag, Herzogenrath, 2018.

[Cambridge Dictionary, 2021] "Variant" on Cambridge Dictionary online. URL: https://dictionary.cambridge.org/dictionary/english/variant (Date: 12.07.2021)

[Chen *et al*., 2001] H. Chen, C. Gomes, B. Selman. Formal Models of Heavy-Tailed Behavior in Combinatorial Search. In: Walsh T. (eds) Principles and Practice of Constraint Programming — CP 2001. CP 2001. Lecture Notes in Computer Science, vol 2239. Springer, Berlin, Heidelberg, 2001.

[Davis *et al*., 1962] Martin Davis, George Logemann, Donald Loveland. A Machine Program for Theorem-Proving. In: Communications of the ACM 5 (7), S. 394–397, 1962.

[Ehrlenspiel *et al*., 2014] Klaus Ehrlenspiel, Alfons Kiewert, Udo Lindemann, Markus Mörtl. Kostengünstig Entwickeln und Konstruieren. Kostenmanagement bei der integrierten Produktentwicklung. 7. Aufl., Springer, Berlin, Heidelberg, 2014.

[Eisenhart, 2002] Maximilian von Eisenhart Rothe. Konzeption und Einführung eines IT-gestützten Produkt-Konfigurationsmanagements für die technische Information in der Automobilentwicklung und –herstellung. Shaker Verlag, Aachen, 2002.

[Eén and Sörensson, 2004] N. Eén, N. Sörensson. An Extensible SAT-solver. In: E. Giunchiglia, A. Tacchella (eds) Theory and Applications of Satisfiability Testing. SAT 2003. Lecture Notes in Computer Science, vol 2919. Springer, Berlin, Heidelberg, 2004.

[Frischen *et al.*, 2019] Christian Frischen, Anastasia Marbach, Florian Tichla, Frank Mantwill. Consistent controlling of variants with the aid of the rule-based complex bill of materials [Durchgängige Variantensteuerung mit Hilfe der regelbasierten Komplexstückliste]. In DFX 2019: Proceedings of the 30th Symposium Design for X, Jesteburg, Germany, 2019.

[Große *et al*., 2011] Daniel Große, Görschwin Fey, Rolf Drechsler. Enhanced Formal Verification Flow for Circuits Integrating Debugging and Coverage Analysis. In: Srikanta Patnaik, Raimund Ubar, Jaan Raik und Heinrich Theodor Vierhaus (Hg.): Design and Test Technology for Dependable Systems-on-Chip: IGI Global (Advances in Computer and Electrical Engineering), S. 119–131, 2011.

[Heina, 1999] J. Heina. Variantenmanagement: Kosten-Nutzen-Bewertung zur Optimierung der Variantenvielfalt. Gabler 1999. Zugl. Cottbus: Brandenburgische TU, Diss., Wiesbaden, 1999.

[Herrmann and Seilheimer, 2002] A. Herrmann, C. Seilheimer. Variantenmanagement. In: S. Albers, A. Herrmann (eds), Handbuch Produktmanagement. Gabler Verlag, Wiesbaden, 2002.

[Herlyn, 1990] Wilmjakob Herlyn. Zur Problematik der Abbildung variantenreicher Erzeugnisse in der Automobilindustrie. Techn. Univ., Diss. VDI Verlag, Braunschweig, 1990.

[Hude, 2020] Marlis von der Hude. Logistische Regression – Ein Prognoseverfahren für die Klassifikationsfragestellung. In: Predictive Analytics und Data Mining, S. 125-135. 2020.

[Kappler *et al*., 2010] Jochen Kappler, Andreas Schütte, Heiko Jung, Dennis Arnhold, Uwe Bracht. Robuste Primär- und Sekundärbedarfsplanung komplexer und variantenreicher Serienprodukte. Integrationsaspekte der Simulation: Technik, Organisation und Personal. Gert Zülch, Patricia Stock (Hrsg.). KIT Scientific Publishing, Karlsruhe, 2010.

[Kübler *et al*., 2010] Andreas J. Kübler, Christoph Zengler, Wolfgang Küchlin. Model Counting in Product Configuration, 2010.

[Küchlin, 2020] Wolfgang Küchlin. Symbolische KI für die Produktkonfiguration in der Automobilindustrie, 2020.

[Marques-Silva and Sakallah, 2003] João P. Marques-Silva, Karem A. Sakallah. Grasp - A New Search Algorithm for Satisfiability. In: Andreas Kuehlmann (Hg.): The Best of ICCAD. Boston, MA: Springer US, S. 73–89, 2003.

[Mohri *et al*., 2001] Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar. Foundation of Machine Learning. The MIT Press, Cambridge, Massachusetts, London, Second edition, 2018.

[Moskewicz *et al*., 2001] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, Sharad Malik. Chaff: Engineering an efficient SAT Solver. In: Jan Rabaey (Hg.): Proceedings of the 38th conference on Design automation - DAC '01. Las Vegas, Nevada, United States. New York, New York, USA: ACM Press, S. 530–535, 2001.

[Petrie, 2012] Charles J. Petrie. Automated Configuration Problem Solving, 2012.

[Sinz *et al*., 2003] Carsten Sinz, Andreas Kaiser, Wolfgang Küchlin. Formal Methods forth Validation of Automotive Product Configuration Data, 2003.

[Stich, 2007] Christoph Stich. Produktionsplanung in der Automobilindustrie. Optimierung des Ressourceneinsatzes im Serienanlauf. Kölner Wissenschaftsverlag, Köln, 2007.

[Voronov, 2013] Alexey Voronov. On Formal Methods for Large-Scale Product Configuration, 2013.

[Wagenitz, 2007] Axel Wagenitz. Modellierungsmethode zur Auftragsabwicklung in der Automobilindustrie. Dissertation. Universität Dortmund, Fachbereich Maschinenbau, 2007.

[Wittek, 2013] Kai Wittek. Standortübergreifende Programmplanung in flexiblen Produktionsnetzwerken der Automobilindustrie. Springer Fachmedien, Wiesbaden, 2013.

[Zagel, 2006] Mathias Zagel. Übergreifendes Konzept zur Strukturierung variantenreicher Produkte und Vorgehensweise zur iterativen Produktstruktur-Optimierung, 2006.