TWO-LEVEL OPTIMIZATION OF PRODUCT FAMILIES: APPLICATION TO A PRODUCT FAMILY OF WATER HOSE BOXES

Rötzer, Sebastian (1);
Le Bourgeois, Martin (2);
Thoma, Dominik (2);
Zimmermann, Markus (1)

1: Technische Universität München, Germany;
2: ID-Consult GmbH, Germany

ABSTRACT

Increasing product complexity and individual customer requirements make the design of optimal product families difficult. Numerical optimization supports optimal design but must deal with the following challenges: many design variables, non-linear or discrete dependencies, and many possibilities of assigning shared components to products. Existing approaches use simplifications to alleviate those challenges. However, for use in industrial practice, they often use irrelevant commonality metrics, do not rely on the actual design variables of the product, or are unable to treat discrete variables. We present a two-level approach: (1) a genetic algorithm (GA) to find the best commonality scheme (i.e., assignment scheme of shared components to products) and (2) a particle swarm optimization (PSO) to optimize the design variables for one specific commonality scheme. It measures total cost, comprising manufacturing costs, economies of scales and complexity costs. The approach was applied to a product family consisting of five water hose boxes, each of them being subject to individual technical requirements. The results are discussed in the context of the product family design process.

Keywords: Product families, Numerical methods, Product modelling / models, Complexity, Optimisation

Contact:
Rötzer, Sebastian Josef
Technische Universität München
Mechanical Engineering
Germany
roetzer@pl.mw.tum.de

Cite this article: Rötzer, S., Le Bourgeois, M., Thoma, D., Zimmermann, M. (2021) 'Two-Level Optimization of Product Families: Application to a Product Family of Water Hose Boxes', in *Proceedings of the International Conference on Engineering Design (ICED21),* Gothenburg, Sweden, 16-20 August 2021. DOI:10.1017/pds.2021.587

1 INTRODUCTION

Due to globalization, companies face strong competition. Companies therefore strive to fulfil the customer needs in the best possible way. As customer needs are not uniform, companies are often forced to provide a wide variety of products. However, offering a large range of products usually goes along with an increasing internal variety of parts and components. Due to the high variety and the associated complexity, costs increase. Therefore, the product designers need to apply new strategies to decrease costs, while maintaining an adequate variety of products for the customers.

One effective strategy is creating product families, in which components can be shared among product variants. The designer's task is to determine which components will be shared among the product variants. In practice this task involves choosing (1) the components to be shared, (2) the product variants that will use the common components and (3) the optimal design values for the components.

While this task might be manageable for smaller product families, it quickly becomes complex with an increasing size of the product family and with increasing dependencies between the parts of the product.

Algorithms numerically optimize product families subject to the costs. However, most algorithms are not universally applicable to all product family design tasks. In the following we compare existing approaches for product family optimization and then present a new approach which closes the gap to a group of optimization problems that we observed as essential for industry practice.

2 RELATED RESEARCH

2.1 Variant management and product family design

Designing a family of products to leverage the effect of component standardization represents an important opportunity for industrial companies, since it can reduce the costs of development, logistics and production (Robertson and Ulrich, 1998). On the other hand, too high standardization rates can reduce the customer satisfaction (Robertson and Ulrich, 1998), or require an over-specification of the components, causing overheads for product functionality and costs (Fujita and Yoshida, 2004). The aim of variant management is therefore to find the right level of commonality for each component to strike a good balance between the beneficial and prejudicial effects of standardization.

Besides qualitative frameworks developed for instance by Franke (1998), Long et al. (2009) or Johannesson et al. (2017), other works provide quantitative approaches for product family design which rely on mathematical models of component commonality and of its influence to enable algorithm-based optimization (Simpson et al., 2006b). The same way as in variant management, product family design aims at finding a good trade-off between commonality and differentiation and at designing the component variants (Simpson et al., 2001).

Within the field of product family design, Simpson et al. (2006a) identify two main categories of product families. In module-based product families, new products are created by combining existing functional modules in different ways while, in scale-based product families, design variables (DV) take different values to scale components and create new products with different performance levels that fulfil their respective requirements. The approach presented in this paper is developed for the design of scale-based product families.

2.2 Approaches for quantitative optimization of scale-based product families

Quantitative optimization approaches provide an objective basis for decision-making and to numerically estimate the performance of a product family. They rely on optimization algorithms that aim at finding a value of a decision vector that minimizes one or several criteria evaluated by an objective function, subject to feasibility constraints. In product family design, the decision vector can include the design variables that characterize the dimensions of the products and a commonality scheme (Simpson, 2006).

To identify the existing approaches, which solve such optimization problems, we conducted a literature study on the field of quantitative product family design using the Scopus database. From the obtained results, we retained only available sources that describe an optimization approach that includes the product design variables and the commonality scheme in their decision vector.

To model the effect of component commonality, many approaches include a Commonality Metric (CM) as well as the product performances in a multi-objective optimization process. The CM is a measure of the degree of standardization of a product family that is assumed to be correlated with the

https://doi.org/10.1017/pds.2021.587 Published online by Cambridge University Press

overall profit of the product family. On the contrary, the cost-model presented by Rötzer et al. (2020b) avoids that hypothesis and directly predicts the impact of standardization on the product family cost.
Existing approaches suggest several ways to reduce the complexity of the considered optimization problem. A possible way is for instance to enforce a component commonality by considering only discrete values of the design variables (Liu et al., 2011). Some approaches assume a restrained commonality by considering standardizing a component for either all or none of the product variants (Simpson and D'Souza, 2004), as opposed to a generalized commonality where a component variant can be shared by subsets of the product variants (Fellini et al., 2006). Prior to the optimization itself, some approaches reduce the pool of candidate commonality schemes using a pre-study to identify the most relevant ones. Fellini et al. (2006) first optimize each product individually and perform a cluster analysis to identify standardizations that are likely to be relevant. Wei et al. (2019) compute the sensitivity of the product performances with respect to each design variables and chose the variables with little influence as standard variables.
Besides the optimization heuristics they use, the optimization algorithms differ in the way they organize the optimization of the commonality scheme and the design variables. Some algorithms separate the optimization on two levels: an upper level, which performs the optimization of the commonality scheme, and the lower level which evaluates each pattern by optimizing the design variables within it. On the contrary, in one-level algorithms, both the commonality scheme and the design variables are optimized at the same time.
Approaches that eliminate candidate solutions, such as Fellini et al. (2006), Chen and Wang (2008), Wei et al. (2019), Simpson and D'Souza (2004), Liu et al. (2011) or Wei et al. (2019) reduce the dimension of the problem, but also exhibit two weaknesses: First, the efficiency of the optimization algorithms used for such reduced problem formulations is not proven. Second, the mathematical modelling of the commonality scheme required to embed it within the decision vector is not suitable for a problem with a generalized commonality.
To model the adverse effect of a too high commonality, Simpson and D'Souza (2004), Fellini et al. (2006), Li and Huang (2009), Khajavirad et al. (2009), Liu et al. (2011) or Chowdhury et al. (2013) perform a multi-objective optimization to maximize the commonality as well as a measure of the product performance. Some of them build a Pareto-front between those objectives (Simpson and D'Souza, 2004; Chen and Wang, 2008; Li and Huang, 2009; Wei et al., 2019), while other aggregate them in a meta-objective function (Khajavirad et al., 2009; Liu et al., 2011; Chowdhury et al., 2013). In both cases, the cost or profit are not explicitly estimated and the decision making therefore heavily relies on human judgement. However, the algorithms used by approaches that rely on the meta-objective function can be suitable for our problem formulation.
Although the approach presented by Fujita and Yoshida (2004), which relies on gradient-based algorithms to optimize the design variables, requires therefore the continuity of all objective and constraint functions. That requirement makes it inappropriate for discontinuous problem statements.

3 METHOD

The goal of our approach is to determine the optimal set of common components for a product family. For the product designer the approach involves three major steps. First, she/he needs to create a model of the problem. This includes (1) a technical model, covering the dependencies between the design variables of the product and its quantities of interest and (2) a parametric cost model. The second step is the optimization itself. Here, the designer applies the optimization algorithm that we present in this paper. In the last step the designer needs to interpret the optimization results which involves deciding on a commonality scheme and choosing the values of the design variables.
The technical model quantifies the technical dependencies between the design variables and the quantities of interest (see Roetzer et al. (2020)). The goal of the cost model is to decide on the optimal design of the product family. Therefore, the cost model should cover all relevant cost effects. However, to reduce the effort for creating the cost model, most approaches only cover certain cost effects: Chowdhury *et al.* (2011) include the material cost as a function of the design variables. Liu *et al.* (2010) account for the effect of increased production volumes on the production cost. Park and Simpson (2008) use activity-based-costing to cover the aspect of complexity cost caused by increased variety. Our experience from consultancy projects shows that decisions towards the optimal level of variety are supported best by combining the following three cost effects simultaneously:

- Manufacturing cost dependent on design values of the components
- Complexity cost dependent on the number of variants
- Economies of scale dependent on the production volume of the components

Those costs depend on the design of the product family. If a component is shared among large shares of the product family, the component will be over-dimensioned, which results in increased manufacturing cost. Complexity cost decrease with a lower number of component variants. And finally, economies of scale result from sharing a component between multiple products, thus increasing its production volume. In our approach we use a parametric cost model presented by Rötzer et al. (2020). It covers these three cost aspects.

The analysis of the optimized commonality scheme requires a representation of the commonality. We use a representation based on binary variables where the value is 1 if a component is shared between two products. Similar representations of the commonality scheme are used by Chowdhury et al. (2013) or Fujita and Yoshida (2004). On one hand, the use of the binary variables makes it easy to evaluate different commonality schemes. On the other hand, the number of variables to store the information increases exponentially with the number of components and products. To decrease the combinatorial challenge, we apply a simplification that is reasonable for scale-based product families: We only consider commonality schemes, in which components are shared between "neighbouring" product variants. This means solutions, in which a component would be only shared for example among the largest and smallest product variants are excluded a-priori from the evaluation. Using this simplification, the combinatorics of sharing components within a product family increases linearly with the number of products - and not exponentially anymore.

As stated in the literature review, product family optimization can be solved as one-level or two-level optimization. We adopt the idea of two-level optimization from Fujita and Yoshida (2004) as it makes it possible to divide the complex optimization problem and solve the sub-problems more efficiently with appropriate algorithms. On the upper level, we search for the optimal commonality scheme, represented by the binary variables. On this level, we use a genetic algorithm, due to its strong ability to solve combinatorial problems. On the lower level, the algorithm determines the optimal values for the design variables of the components for a given commonality scheme.

The optimization problem can be formulated as follows, where $\boldsymbol{\xi}$ represents the current commonality scheme passed from the upper to the lower level, $\boldsymbol{x}$ the DVs of the components of the product family, $\boldsymbol{g}(\boldsymbol{x})$ the technical requirements on the product, and C_F the total cost of the product family:

$$\min_{\boldsymbol{\xi}} f(\boldsymbol{\xi}, \boldsymbol{x}) = C_F \qquad \min_{\boldsymbol{x}} f(\boldsymbol{\xi}, \boldsymbol{x}) = C_F$$

$$\text{Subject to } \boldsymbol{\xi} \in \{0; 1\} \qquad \text{Subject to} \begin{cases} \boldsymbol{g}(\boldsymbol{x}) \leq 0 \\ \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{\xi}) = 0 \\ \boldsymbol{x_l} \leq \boldsymbol{x} \leq \boldsymbol{x_u} \end{cases}$$

While the design variables $\boldsymbol{x}$ are varied on the lower level, the binary commonality variables $\boldsymbol{\xi}$ are varied on the upper level and are fixed on the lower level. To further reduce computational effort, we do not explicitly enforce a specific commonality scheme on the lower level by an equal constraint function $\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{\xi})$, but directly adopt the number of design variables used according to the specific commonality schemes. Thus, less function evaluations are necessary and the equal constraint function does not need to be ensured. Figure 1 depicts our algorithm to solve the optimization problem formulated above.

For the lower level optimization, Fujita and Yoshida use sequential quadratic programming (SQL), which works for continuous problems. However, the experience from industry projects shows, that many industry problems involve discontinuities of the optimization problem. For this reason, we saw the necessity to choose an algorithm that can handle discontinuous functions. While the MATLAB implementation of the genetic algorithm was not able to find results in our test, particle swarm optimization (PSO) yielded good results. Other works on product family optimization also use PSO as the optimization algorithm (Wang, 2011; Chowdhury *et al.*, 2013). We think that also other heuristic algorithms like simulated annealing or ant colony optimization could prove successful in this task. Our approach allows the exchange of the used algorithm. Due to the stochastic nature of PSO the results of the optimization of the design variables are subject to variance. This means, that the results

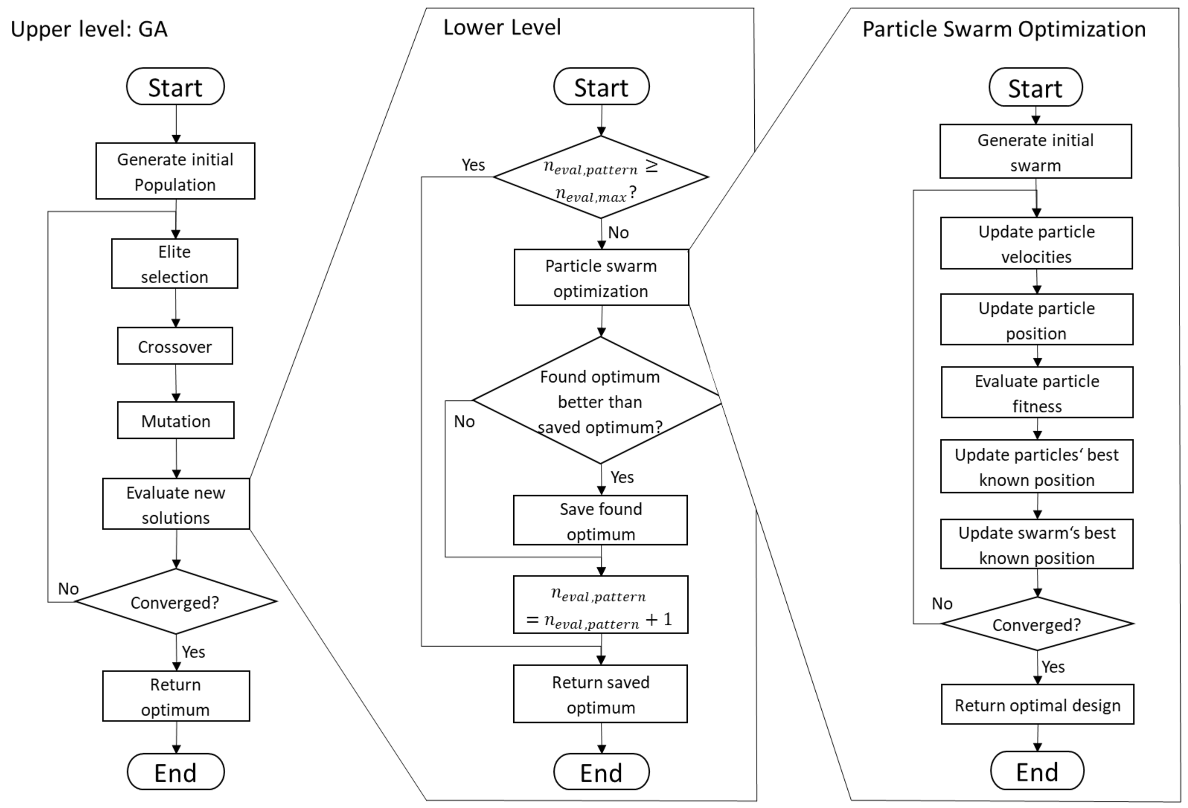

Upper level: GA
Start
Generate initial Population
Elite selection
Crossover
Mutation
Evaluate new solutions
No
Converged?
Yes
Return optimum
End
Lower Level
Start
Yes
$n_{eval,pattern} \geq n_{eval,max}$?
No
Particle swarm optimization
Found optimum better than saved optimum?
No
Yes
Save found optimum
$n_{eval,pattern} = n_{eval,pattern} + 1$
Return saved optimum
End
Particle Swarm Optimization
Start
Generate initial swarm
Update particle velocities
Update particle position
Evaluate particle fitness
Update particles' best known position
Update swarm's best known position
No
Converged?
Yes
Return optimal design
End

Figure 1. Flow chart of the two-level product family optimization (TL-PFO) algorithm. Upper level with GA and lower level with PSO

differ from one run to another. To cope with this problem, the lower level optimization is carried out several times for each commonality scheme and the best result is taken. The number of evaluations for each pattern is a parameter of the approach. The higher the parameter, the lower the impact of the variance. In return, the computational effort increases with the number of evaluations.

During an optimization run, the GA on the upper level might generate the same commonality scheme twice or more in different generations. In this case, the evaluation of this commonality scheme on the lower level would have to be carried out repeatedly. To avoid this unnecessary effort, we safe the results of the lower level evaluation. So, the second time an individual appears, the algorithm will use existing results without additional computation.

The process of determining an initial population for the PSO has a strong impact on the computation time of the algorithm and even influences on the quality of the results. Our approach follows this logic: First, we build a latin hypercube sample (LHS) of the design space. The size of the hypercube equals the size of the initial swarm. Samples, that violate the constraint functions, are modified by solving a multi-objective goal attainment problem. If some samples cannot be turned into good designs, they will be replaced by generating a new LHS. In the case, that in the first round all samples are invalid and none can be turned into a good sample, we assume that no solution exists for the currently analysed commonality scheme. Then a new commonality will be determined by the GA on the upper level. Figure 1 shows the steps our two-level product family optimization (TL-PFO) approach and visualizes the interaction between upper and lower level.

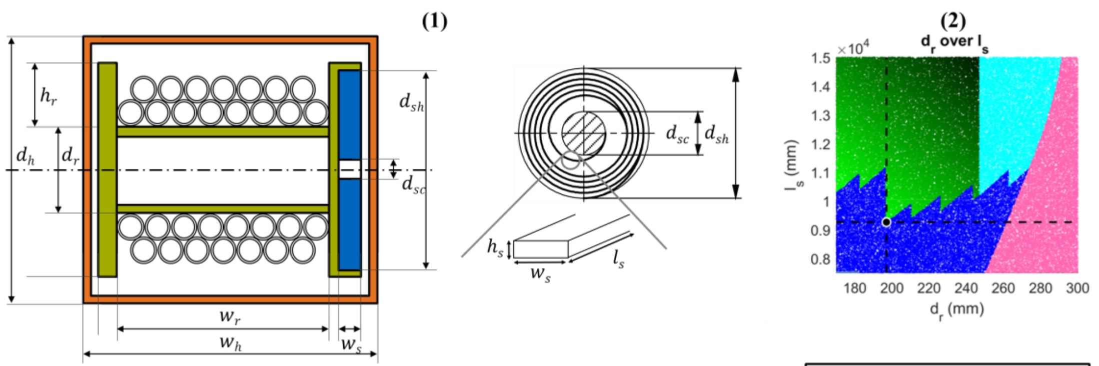
4 RESULTS

We applied TL-PFO on an example from an industry project of a consulting company. First, a statistical evaluation performed on many optimizations for reduced product families using different sets of cost parameters made it possible to check the plausibility of the results. Then, we discuss the results of TL-PFO using the complete product family.

4.1 Application example: a family of garden hose boxes

The considered application example consists of a family of water hose boxes composed of three components each: a spool where the hose is rolled up, a cylindric housing and a spring to automatically

https://doi.org/10.1017/pds.2021.587 Published online by Cambridge University Press



d_r	Outer diameter reel center	h_s	Height spring strip
w_r	Inner width reel	w_s	Width spring strip
h_r	Height reel jaw	l_s	Length spring strip
w_h	Outer width housing	d_{sh}	Diameter spring housing
d_h	Outer diameter housing	d_{sc}	Diameter spring core

● Feasible solutions
● Violating y_2: min. insertion force
● Violating y_3: max. insertion force
● Violating y_6: gap between reel-housing diameters

Figure 2. (1) design variables of the hose box, (2) Projection of feasible and unfeasible hose box designs; blue spikes indicate discontinuous dependencies

roll in the hose (see Figure 2). A particular box variant is defined by ten design variables and ten quantities of interest that are deduced from the design variables using the mathematical model developed by Rötzer et al. (2020). The box variants differ from one another according to the hose length they must store: 15m, 20m, 25m, 30m and 35m.

Figure 2 shows the solution space for a hose box variant. In this case two design variables illustrate a discontinuous effect: d_r is the diameter of the spool where the hose is rolled up; l_S: length of the spring strip. The relevant requirement is the minimum insertion force (y_4), which is necessary to roll up the hose automatically. l_S (together with other design variables) determines the available torque of the spring. d_r (together with other design variables) determines the radial position of the hose. The torque of the spring together with maximum radial position of the hose (lever) defines the minimum insertion force available, which is compared against requirement y_4. If the minimum insertion force is too small (y_4 violated), it is indicated by the blue area in the diagram. We can see that increasing d_r lead to lower minimum insertion forces and thus violate the requirement y_4. Increasing l_S increase the available torque and thus, can compensate this effect. Increasing d_r also increase the maximum length which can be rolled up by the spool. At certain value we can save a radial hose layer. Then the lever decreases, and the available minimum insertion force increases by leaps and bounds. These effects induce spikes that create local optima, which cause problems for gradient-based optimization algorithms – and also for designers, as they are highly non-linear.

An optimization algorithm, and more likely a gradient-based algorithm, might get stuck in those spikes while looking for the most cost-efficient design. This reduces the quality and the repeatability of the results. As a result, the performed plausibility check also aims at assessing the impact of those local optima on the performance and robustness of the algorithm.

4.2 Plausibility check of the algorithm

We restricted the product families to the two smallest box variants to reduce computational effort. We defined three cost scenarios: in the pessimistic scenario, the economies of scales and complexity costs are zero and component commonality has therefore no effect on the total costs while, on the contrary, it is very beneficial in the optimistic scenario. In the realistic scenario, it has a moderate impact on the costs. For each cost scenario, we performed 100 optimizations of a family including only the 15m and 20m box and 50 optimizations of a family including the three smallest boxes (15m, 20m and 25m). Figure 3 shows the costs obtained by the optimization runs for the 15m and 20m box. For the pessimistic and realistic scenarios, the relative standard deviation of the obtained total costs amounts respectively 1.32% and 2.47%. For the optimistic scenario, its value reaches 10.86% due to a small number of obtained costs laying above the mean value. This phenomenon illustrates the possible impact of local optima on the results and the robustness of the algorithm.

Although the stochastic behaviour of the used algorithm also leads to a variability of the obtained commonality schemes, a detailed analysis shows that the variation of the parameter values has the expected impact on the results of the algorithm.

https://doi.org/10.1017/pds.2021.587 Published online by Cambridge University Press

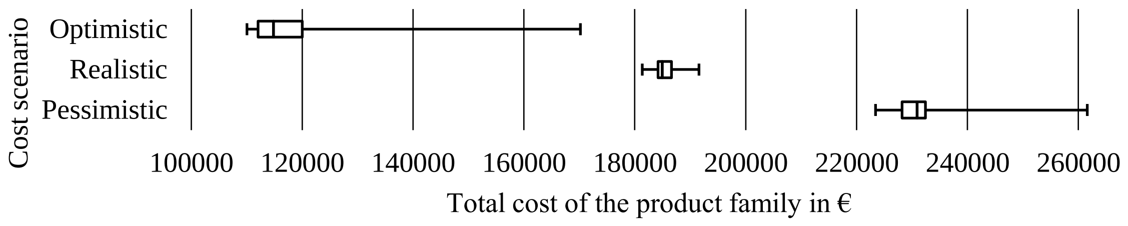


Figure 3. Costs obtained by optimizing a family of two hose boxes for three different cost scenarios (i.e., sets of cost parameters) depicted as box plots.

The mean value of the obtained product family cost is lower in the optimistic scenario than in the pessimistic scenario due to the important savings made possible by component commonality. In addition, the optimized product families exhibit a higher commonality if the economies of scale and complexity costs are larger. In the optimistic scenario, for instance, 84 out of the 100 optimizations suggested a fully standardized family. On the contrary, for the pessimistic scenario, the most frequent commonality scheme includes no commonality at all. However, that pattern only appears in 37 out of 100 results. In that case, the repeatability of the results is negatively affected by the existence of different designs of the product family that exhibit similar costs. More generally, the possible compensation of oversizing by component commonality as well as the interaction between the components, for instance if a larger spool also requires building a larger housing, create local optima that reduce the repeatability of the results. The optimizations performed on the family of three products lead to similar observations. The total product family cost for the optimistic, realistic, and pessimistic scenario varies respectively 3.24%, 2.09% and 1.14%. In that case, component standardization rate also increases with higher economies of scales and complexity costs.

4.3 Optimization of the complete product family using TL-PFO

After a statistical evaluation of the results, we use the TL-PFO approach to optimize the complete family of five boxes. Due to the higher computational effort, five optimizations are performed using TL-PFO. Table 1 shows the result of five optimization runs. The observations resemble the statistical evaluation of the approach. The lowest and highest costs obtained by an optimization (run 2 and 4 respectively) exhibit a deviation from the average result of respectively 1.7% and 2.3%. The obtained commonality schemes differ from one another. However, despite that variability, some similarities between them provide valuable indications to support decision-making. Figure 4 illustrates the resulting product families with their corresponding component variants for the five optimization runs.
For instance, run 2 - 4 include highly standardized housing with only one variant. While, on the contrary, the spring tends to be strongly differentiated in all optimization runs. The spool also tends to standardization with one component (run 2) or two components (run 1, 3, 4). Run 5 suggests a highly differentiated product family with individual components for nearly each product variant. All results are optimized product families. Due to the complexity of the problem different solutions can lead to similar costs: Run 2 reduces complexity costs by standardization; run 5 reduces costs by designing the components specifically for each product, thus no over-dimensioning is needed, and manufacturing

Table 1. Commonality schemes of the optimization of the family of garden hose boxes and their objective function value.

C1: Spool C2: Housing C3: Spring	Run 1			Run 2			Run 3			Run 4			Run 5			
	C1	C2	C3	C1	C2	C3	C1	C2	C3	C1	C2	C3	C1	C2	C3	
15m Box	1.1	2.1	3.1	1.1	2.1	3.1	1.1	2.1	3.1	1.1	2.1	3.1	1.1	2.1	3.1	
20m Box	1.1	2.1	3.2	1.1	2.1	3.2	1.1	2.1	3.2	1.2	2.1	3.2	1.2	2.2	3.1	
25m Box	1.1	2.1	3.3	1.1	2.1	3.3	1.1	2.1	3.3	1.2	2.1	3.3	1.3	2.3	3.2	
30m Box	1.2	2.2	3.4	1.1	2.1	3.4	1.2	2.1	3.4	1.2	2.1	3.4	1.4	2.4	3.3	
35m Box	1.2	2.2	3.5	1.1	2.1	3.5	1.2	2.1	3.4	1.2	2.1	3.5	1.5	2.5	3.4	Average cost:
Cost:	647,688 €			639,555 €			641,990 €			656,554 €			641,572 €			645,472 €

https://doi.org/10.1017/pds.2021.587 Published online by Cambridge University Press

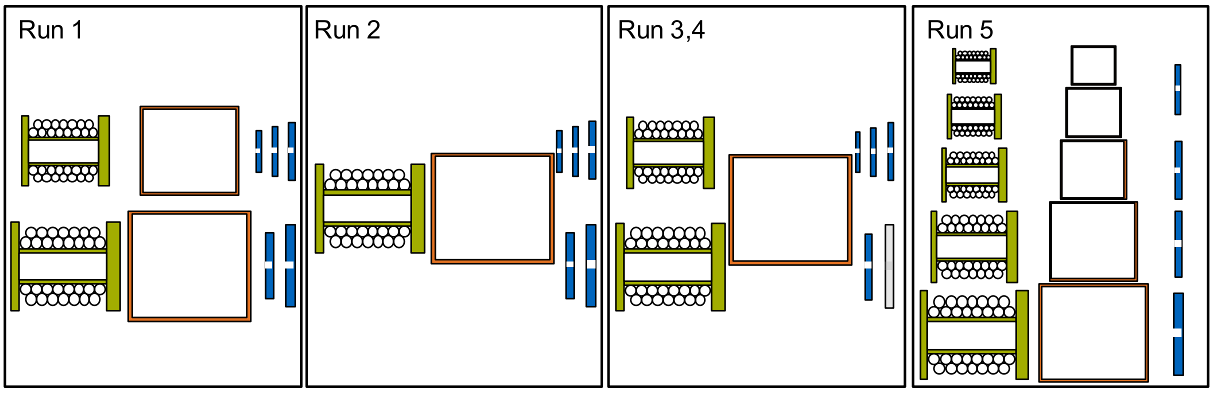


Figure 4. Product families with their component variants resulting from optimization results from Table 1

costs can be saved. Both effects lead to similar overall costs in this example.
TL-PSO makes it possible to find an optimized design for the considered product family despite the large number of possible commonalities. In the considered example with five boxes made of three components each, the total number of possible commonality schemes outreaches 140.000, and is reduced to 4096 by the assumption of sharing components between neighbouring products. Within each commonality scheme, the optimization problem to be solved can include between 3 (if all components are standard) and 50 (if there is no commonality) design variables. This leads to computation time of, on average, 8 h 48 min to optimize a family of five boxes. For larger product families, an extrapolation foresees for instance over 47 h to optimize a family of seven products.

5 SUMMARY AND DISCUSSION

In this paper we present a method to optimize a product family with respect to its costs. The suggested method is based on a literature review. Among the existing approaches we could not find an optimization algorithm, which can be applied to the problem at hand. The problem originates from an industry project at a consulting company. A family of five water hose boxes needs to be optimized. It consists of three main components. They are described by ten design variables. The product is described by ten quantities of interests and their variant-specific requirements. Challenges arose from discrete constraint functions, a high number of evaluations due to combinatorics, and the cost modelling of the objective function. Motivated by this industry example, we created a new approach by combining different ingredients from existing approaches. The result is a two-level optimization algorithm: on the first level the commonality scheme is optimized, the second level algorithm optimizes the costs of one specific commonality scheme with respect to its technical requirements. Thus, the second level optimization provides input for the first level. For the first level optimization we use a genetic algorithm due to the binary structure of commonality schemes. On the second level we use particle swarm optimization (PSO) to deal with discontinuous constraint functions. The PSO provides a cost optimized set of design variables for a given commonality scheme subject to the fulfilment of the technical requirements on the product family. To alleviate the high number of function evaluations, we used three measures: (1) We only allow commonality between neighboured product variants, which is reasonable for a scale-based product family. (2) We use a genetic algorithm for the optimization of the commonality scheme. Already optimized commonality schemes are stored and can be re-used directly in the genetic algorithm. (3) We do not explicitly enforce a specific commonality scheme on the lower level by an equal constraint function, but directly adopt the number of design variables used according to the specific commonality As an objective function we used a model that includes manufacturing costs (dependent on the design variables $\boldsymbol{x}$); multiplied by a factor to consider enconomies of scale (dependent on the sales volumes of the different product variants n_P); and complexity costs (dependent on the number of component variants n_C). The result of the algorithm is a cost optimized commonality scheme with its design variable values and overall product family cost.
Due to the stochastic algorithms, the results are subject to variance. In order to check both plausibility and repeatability of the results, we conducted a study with 100 optimization runs for two product variants and 5 runs for five product variants. Within the optimization runs all parameters are fixed. For optimistic cost scenarios we get patterns with high commonality. For pessimistic scenarios, the algorithm

tends to individual optimization of the components according to their manufacturing costs. The results are thus plausible. Whereas the values of the objective function are subject to comparatively small deviations, the commonality schemes deviate more from each other. This can be explained by looking at the pessimistic scenario: the most likely pattern (a unique component for each product) appeared in only 37/100 runs, but the costs deviate only by approx. 1%. When we look at the results, we can find many local minima with very similar cost values. In realistic scenarios positive effects of standardization for one component can lead to negative effects for another component (e.g. bigger spool leads to bigger housing). Thus, different commonality schemes can lead to similar objective function values.
Due to combinatorics the designer would have to evaluate more than 140000 possibilities. Furthermore, the designer needs to check that the requirements of all product variants are met and calculate the overall cost. The proposed method provides alternatives with minimized costs to the designer. She/He can then choose the best-fitting alternative for the problem. She/He can also augment the decision process with implicit information, which can hardly be included in algorithms (such as product differentiation on the market). For the family of water hose boxes, we get five different optimized alternatives (see Figure 4). Run 2, 3 and 4 propose one standardized housing. Thus, customers cannot immediately identify the different product variants. The different spools in run 3 and 4 are not visible. Run 1 suggests two housing variants. Run 5 completely differentiates the product family. From a customer's perspective this is the most transparent alternative. Assuming increasing sales volumes run 5 would have disadvantages compared to an alternative with higher commonality. According to this criterion, run 2 is best. The company can differentiate a product family according to run 2 with the different springs. As they have higher manufacturing costs due to higher material costs, this is reasonable. Furthermore, springs could be semi-standardized by e.g. using the same values for the thickness and the width of the spring steel and varying the lengths of the spring steel and the diameters of the spring. Therefore, we would suggest the alternative from run 2 in this specific example.
With respect to the calculation time the algorithm provides results in an adequate time for relatively small numbers of product variants. It increases exponentially with higher numbers of product variants. Considering the need to run an optimization several times due to the stochastic deviation, this can be a major drawback for more complex products with more components and product variants. Furthermore, the modelling of the technical system and the costs induces high efforts.

6 CONCLUSION AND OUTLOOK

Our approach addresses the challenges that arise from industry applications. It solves both an assignment and a cost optimization problem subject to technical constraints. There are no limitations on the constraint functions. The method needs quantifiable requirements, dependencies, and cost effects. In this use case, the algorithm optimizes a product family with five product variants, ten design variables, ten quantities of interest in an adequate time of about 530min. Systematic tuning of the parameters of the optimization algorithm may alleviate the high computational times. To reduce the modelling effort we suggest the automated modelling by Rötzer *et al.* (2020a). To cope with an increasing number of product variants and design variables a combination with the Solution Space Engineering approach for product families might be beneficial (Rötzer *et al.*, 2020b). It reduces complexity by decoupling design variables from each other. The decoupling comes along with a loss of solution space. Thus, the trade-off between computational time and optimality is a focus for future research.

ACKNOWLEDGMENTS

We want to thank Lukas Krischer for the discussions on optimization algorithms and his valuable input.

REFERENCES

Chen, C. and Wang, L. (2008), "A modified genetic algorithm for product family optimization with platform specified by information theoretical approach", *Journal of Shanghai Jiaotong University (Science)*, Vol. 13 No. 3, pp. 304–311.

Chowdhury, S., Messac, A. and Khire, R. (2013), "Investigating the commonality attributes for scaling product families using comprehensive product platform planning (CP3)", *Structural and Multidisciplinary Optimization*, Vol. 48 No. 6, pp. 1089–1107.

Chowdhury, S., Messac, A. and Khire, R.A. (2011), "Comprehensive Product Platform Planning (CP3) Framework", *Journal of Mechanical Design*, Vol. 133 No. 10.

https://doi.org/10.1017/pds.2021.587 Published online by Cambridge University Press

Fellini, R., Kokkolaras, M. and Papalambros, P.Y. (2006), “Quantitative platform selection in optimal design of product families, with application to automotive engine design”, *Journal of Engineering Design*, Vol. 17 No. 5, pp. 429–446.

Franke, H.-J. (1998), “Variantenvielfalt und Resultierende Komplexität. Ursachen und Methoden zu Ihrer Bewältigung”, paper presented at 9th Symposium on Design for Manufacturing, 15.-16.10.1998, Schnaittach/Erlangen.

Fujita, K. and Yoshida, H. (2004), “Product Variety Optimization Simultaneously Designing Module Combination and Module Attributes”, *Concurrent Engineering*, Vol. 12 No. 2, pp. 105–118.

Johannesson, H., Landahl, J., Levandowski, C. and Raudberget, D. (2017), “Development of product platforms: Theory and methodology”, *Concurrent Engineering*, Vol. 25 No. 3, pp. 195–211.

Khajavirad, A., Michalek, J.J. and Simpson, T.W. (2009), “An efficient decomposed multiobjective genetic algorithm for solving the joint product platform selection and product family design problem with generalized commonality”, *Structural and Multidisciplinary Optimization*, Vol. 39 No. 2, pp. 187–201.

Li, L. and Huang, G.Q. (2009), “Multiobjective evolutionary optimisation for adaptive product family design”, *International Journal of Computer Integrated Manufacturing*, Vol. 22 No. 4, pp. 299–314.

Liu, Z., Wong, Y.S. and Lee, K.S. (2010), “Modularity analysis and commonality design: a framework for the top-down platform and product family design”, *International Journal of Production Research*, Vol. 48 No. 12, pp. 3657–3680.

Liu, Z., Wong, Y.S. and Lee, K.S. (2011), “A manufacturing-oriented approach for multi-platforming product family design with modified genetic algorithm”, *Journal of Intelligent Manufacturing*, Vol. 22 No. 6, pp. 891–907.

Long, D., Seering, W. and Rebentisch, E. (2009), “Finding Opportunities for Commonality in Complex Systems”, in Norell Bergendahl, M. (Ed.), *Product and systems design*, Design Society, Glasgow.

Park, J. and Simpson, T.W. (2008), “Toward an activity-based costing system for product families and product platforms in the early stages of development”, *International Journal of Production Research*, Vol. 46 No. 1, pp. 99–130.

Robertson, D. and Ulrich, K. (1998), “Planning for Product Platforms”, *Sloan Management Review*, Vol. 39 No. 4, pp. 19–31.

Rötzer, S., Rostan, N., Steger, H.C., Vogel-Heuser, B. and Zimmermann, M. (2020a), “Sequencing of Information in Modular Model-based Systems Design”, *Proceedings of the 22nd International DSM Conference. Cambridge, Massachusetts, USA, October, 2020 13th-15th 2020. in press*.

Rötzer, S., Thoma, D. and Zimmermann, M. (2020b), “Cost Optimization of Product Families Using Solution Spaces”, *Proceedings of the Design Society: DESIGN Conference*, Vol. 1, pp. 1087–1094.

Simpson, T.W. (2006), “Methods for Optimizing Product Platforms and Product Families. Overview and Classification”, in Simpson, T.W., Siddique, Z. and Jiao, J. (Eds.), *Product platform and product family design: Methods and applications*, Springer, New York, London, pp. 133–156.

Simpson, T.W. and D’Souza, B.S. (2004), “Assessing Variable Levels of Platform Commonality Within a Product Family Using a Multiobjective Genetic Algorithm”, *Concurrent Engineering*, Vol. 12 No. 2, pp. 119–129.

Simpson, T.W., Seepersad, C.C. and Mistree, F. (2001), “Balancing Commonality and Performance within the Concurrent Design of Multiple Products in a Product Family”, *Concurrent Engineering*, Vol. 9 No. 3, pp. 177–190.

Simpson, T.W., Siddique, Z. and Jiao, J. (2006a), “Platform-Based Product Family Development. Introduction and Overview”, in Simpson, T.W., Siddique, Z. and Jiao, J. (Eds.), *Product platform and product family design: Methods and applications*, Springer, New York, London, pp. 1–15.

Simpson, T.W., Siddique, Z. and Jiao, J. (Eds.) (2006b), *Product platform and product family design: Methods and applications*, Springer, New York, London.

Wang, W. (2011), “Scalable platform design optimization using hybrid co-evolutionary algorithms”, in Staff, I. (Ed.), *2011 IEEE 2nd International Conference on Software Engineering and Service Science, 7/15/2011 - 7/17/2011, Beijing, China*, IEEE, [Place of publication not identified], pp. 270–273.

Wei, W., Tian, Z., Peng, C., Liu, A. and Zhang, Z. (2019), “Product family flexibility design method based on hybrid adaptive ant colony algorithm”, *Soft Computing*, Vol. 23 No. 20, pp. 10509–10520.

https://doi.org/10.1017/pds.2021.587 Published online by Cambridge University Press