INTERNATIONAL DESIGN CONFERENCE – DESIGN 2020
https://doi.org/10.1017/dsd.2020.283

A MULTI-DIMENSIONAL CONFIGURATION ALGORITHM FOR MODULAR PRODUCT ARCHITECTURES

F. M. Seiler ✉ and D. Krause

Hamburg University of Technology, Germany

✉ florian.seiler@tuhh.de

Abstract

With an increasing demand for product individualisation leading to increased product architecture complexity and -costs, modular kits are one common measure to cope with this issue. The management of such a modular kit as well as the methodical determination of a specific product variant is key to the manufacturer's success. As multiple influence factors need to be taken into account when configuring product variants, we propose a multi-dimensional geometric optimisation algorithm, allowing for prioritising varying customer demands and thereby determining the ideally balanced product variant.

Keywords: configuration management, modularisation, product architecture, constraint modelling, model-based engineering

1. Introduction

With today's major trends on the global market, a constant competition and specialisation within production companies can be noted, resulting in an ever-increasing demand for highly individualised products. At the same time, cycle times and production costs need to be as low as possible. In order to cope with this at first glance contradictory situation, the concept of modularisation provides one possible solution to this problem by reducing the internal variety whilst keeping external variety still high (Krause and Gebhardt, 2018).

There are several different modularisation methods, which all result in a modular product architecture (Simpson et al. 2006; Erixon 1998; Krause and Gebhardt, 2018; etc.). As the individual modules are linked with or without individual restrictions amongst each other, a set of rules for the individual modular architecture of modular product families can be derived. This set of rules combined with the modular product architecture can be regarded as a modular construction kit (Bursac, 2016).

This set of rules and constraints increases exponentially with the size and complexity of the modular architecture. From a certain size of this modular construction kit on, the set of constraints and dependencies is no longer manageable without the help of an adequate software solution (Hermann et al., 2013). In order to develop such a software solution, the modular kit and most important the set of dependencies need to be transformed into an appropriate, software-useable shape. As modular architectures and those dependencies are built from different data types and have a complex linking structure underlying, a regular database such as SQL or NoSQL-Databases does not meet the required needs. As we have shown, the concept of model-based systems engineering (MBSE) provides a suitable solution for modelling these modular product architectures with their underlying set of dependencies (Seiler et al., 2019a).

Based upon this modelled architecture, a configuration software solution for choosing the right product variant (or configuration), which corresponds the most with the individual customer's needs can be created. With customer demands leading in the most cases not to only one possible solution for a corresponding product configuration, the configuration problem can be regarded as a multi-factor problem. The major issue therefore is to balance all different customer's demands against the capabilities and performance of the individual modules in order to determine not only a possible but the under the underlying circumstances optimal solution. Therefore, not only all customer's demands concerning the product's performance are to be considered, but also the customer's priorities on external influences, such as delivery time or product flexibility. As Weisz et al. already stated, such problems consist of a non-descript system of equation with a varying number of non-descript variables, this solution requires a non-trivial optimisation algorithm (Weisz et al., 2018). This contribution proposes a mathematical approach to this problem by regarding the issue as a multi-dimensional problem in a multidimensional hyperspace and solving it via an infinitesimal minimalization algorithm.
At first, an overview about the state of the art concerning configuration algorithms and multi-dimensional optimisation problems is presented in section 2. Secondly, the proposed hyperspace algorithm is explained in section 3, followed by a case study of an exemplary configuration problem, which has been performed within a german special equipment manufacturer.

2. State of the art

In the following, the state of the art concerning methods and algorithms for such configuration problems is presented.
There already exist a vast amount of different configuration algorithms, which mainly can be divided into two major fields, which are on the one hand "option-based algorithms" and on the other hand "target-figure oriented algorithms" (Liebisch, 2014). Target-figure oriented algorithms clearly check the modular kit with the goal of meeting all required demands, beginning with the simple, sequential reading and checking of all constraints of the modular kit (Stormer, 2007). As every constraint is regarded, no matter if it is relevant for the current problem or not, this solution results in very long calculation times and inhibits the consideration of partial problem solutions.
In order to cope with this issue, the first model-based algorithms were developed. Enabling the modelling of complex and consecutively built structures, decision trees for individual configuration problems can be calculated recursively in advance. With the underlying structure basically being built up as a tree chart, the possible branches of the tree and therefore the amount of dependencies and constraints, which need to be considered can be decreased step by step with every level moved down in the tree chart. (Hadzic and Andersen, 2006). This general approach can be as well performed as a target-figure oriented algorithm, which performs very efficiently if there is a perfect solution for all customer demands. Krieter et al. describe a possible algorithm for such a configuration problem are by proposing a propagating decision method, where every knot within the tree chart leads to a mandatory or optional decision. By implementing a cross-branch advance check of decision consequences, they further reduce the amount of constraints to be checked and therefore increase the configuration algorithm's performance (Krieter et al., 2019).
If the modular kit only allows for a partial fit, this algorithm needs to be designed as an option-based algorithm. Thereby, all modules within the modular kit are considered as options, which can be drawn into the product variant piece by piece for every independent customer demand. This leads to a bunch of possible partial solutions with a varying degree of demand satisfaction, as this option-based algorithm does not require a product variant to meet all demands but also provides solutions, which in the least case only meet one single independent customer demand (Zennaro et al., 2019).
Cicconi et al. (2018) propose such an approach by using model-based structure simulations in order to determine possible solutions for individual customers' demands. They use the customer's demands as the simulation framework and therefore determine different possible solutions within the modular architecture (Cicconi et al., 2018).
The major issue of these kinds of algorithms is, that all these derived solutions are regarded as equally performant and can only be distinguished by the apparent degree of demand satisfaction. As it is highly likely that a modular product architecture will not correspond with all individual customers'

demands and the demand priorisation amongst different customers does not correspond, we consider this degree of demand satisfaction alone as potentially misleading. In order to determine the most appropriate product variant for an individual customer, not only the customer's demands but as well as their priority needs to be considered and balanced together with the performance of the available modular kit.

3. The hyperspace algorithm

As described above, such a multi-variate problem cannot purely be addressed via a system of equations, but needs to be addressed via an optimization algorithm. As all influence factors are strongly dependant and therefore constantly changing the possible solution space with each iteration, a pure algebraic solution, e.g. via a system of differential equations would come with a large calculation effort. Especially when applying the algorithm to different modular architectures for different products or to the same modular architecture but with different priorisation indices, a large adaption effort would be needed. In order to cope with this problem, we propose a geometric solution, basically resulting in combining and minimizing joint multi-dimensional areas by integrating dimension-wise twisted spatial areas over an ideally balanced solution.

The first step is to identify all possible solutions as well as all partial solutions for the individual customers' needs based upon the available modular architecture. In order to cope with the model-based structure of the modular architecture and the dependency set, we use an adapted Dijkstra-algorithm, a mathematical approach usually used within road navigation systems. As an analogy, the modular product architecture can be seen as a "roadmap", with cities representing customer relevant product characteristics and roads and intersections representing modules with dependencies and constraints. The Dijkstra-algorithm is a so called "greedy-algorithm" and was originally designed to determine the shortest path between two points. Further adaptions increased the amount of points, between which possible paths and the lengths of these paths are to be calculated, to an infinite amount (Dijkstra, 1959; Broumi, 2019). The following Figure 1 shows an according to the above described analogy adapted scheme, which is used for the determination of the suitable product variants, including partial solutions. To make the algorithm more understandable, Figure 1 is reduced to a 2D-perspective, the calculation basis is represented in an n-dimensional space.

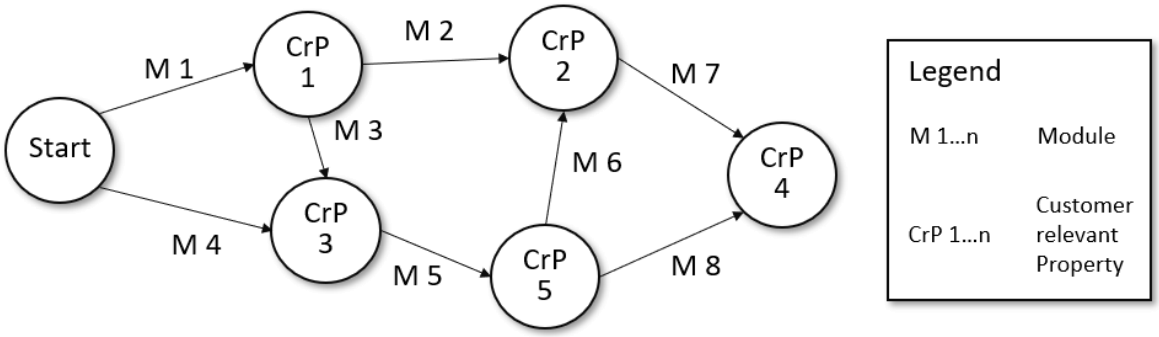

Figure 1. Modular road map scheme for Dijkstra-algorithm

In order to calculate variants and determine knots and path lengths from this structure, the solution space of the modular product architecture gets transformed into a symmetrical adjacency matrix. The only restriction for such an adjacency matrix is $n \in [0; \infty [$ with $\boldsymbol{n}$ representing the number of knots. The values in this adjacency matrix represent the path length to the corresponding next knot. In order to consider all existing dimensions (which can be seen as multiple, connected layers of the "modular roadmap") for the calculation of the individual knot-to-knot distance, there needs to be one independent adjacency matrix for every combination of dimensions with the single use of every dimension without consideration of their order, as adjacency matrices are symmetrical. The number of required adjacency matrices is calculated trough the following binomial coefficient with $\boldsymbol{n}$ representing the dimension number (Equation 1). These multiple-layered adjacency matrices are to cope with the multiple customer-relevant properties to be reached by one module, as just one matrix would only allow for a module to be linked to two customer

relevant properties in maximum. By implementing multiple matrices with connected dimensions, the amount of customer-relevant properties fulfilled by one module is not limited any more.

$$\binom{n}{2} = \frac{n!}{(n-2)!*2!} \tag{1}$$

Figure 2 shows the corresponding adjacency matrix for Figure 1.

	$Start$	$CrP\,1$	$CrP\,2$	$CrP\,3$	$CrP\,4$	$CrP\,5$
$Start$	0	$M1$	0	$M4$	0	0
$CrP\,1$	$M1$	0	$M2$	$M3$	0	0
$CrP\,2$	0	$M2$	0	0	$M7$	$M6$
$CrP\,3$	$M4$	$M3$	0	0	0	$M5$
$CrP\,4$	0	0	$M7$	0	0	$M8$
$CrP\,5$	0	0	$M6$	$M5$	$M8$	0

With this adjacency matrix as a calculation basis, a software can "navigate" to each customer demand and map the possible paths towards these demands. By storing the possible paths in an intermediate database, all relevant modules of each complete set of paths can be derived, defining the definition range $\mathbb{D}$ for this configuration.

With all possible paths k_i *with* $k \in \mathbb{D}$ derived, the second step of the algorithm is the optimisation step. As described above, this optimisation process is based upon a geometric solution. The solution space of the configuration problem is regarded as a multi-dimensional space, which is usually referred to as "hyperspace" (Diestel, 2000). The dimensions of this hyperspace are the individual customer-relevant objectives within the individual project, such as delivery time, total cost, efficiency, sustainability, material-to-workload ratio, spare parts availability, overall equipment effectiveness, system performance, user-friendliness, level of system engineering completeness etc. As all dimensions originally come with different units or even are not scalable in absolute numbers, the first step when creating the hyperspace is to normalize all dimensions, which then results in a dimensionless hypercube, (Schläfli-Symbol $\{4, 3^{d-2}\}$) where ***d*** is the dimension number or equally the number of customer-relevant objectives within a modular product architecture. The space diagonal of this hypercube represents the ideally balanced solution for the individual configuration problem with the ***Idealpoint S*** at the normalized ending edge of the n-dimensional hypercube as an ideally balanced objective with a maximum expression in each dimension. This Idealpoint S would represent a product variant exactly meeting all customer demands to full extent and providing suitable modules from the modular kit.

As a next step, all modules for each individual product variant determined via the above described adapted Dijkstra-algorithm then get mapped within this hypercube. At this point, the above noted priorisation of an individual customer for an individual customer-relevant objective is introduced into the algorithm. As each module contains a value for each dimension of the hypercube (representing the customer-relevant objectives), these values are multiplied with a scalar prioritising number ***p***, also determining the final edge length ***a*** of the hypercube, with the following conditions:

$$\begin{Bmatrix} p=0 & standard \\ p=1 & slight\,emphasis \\ p=2 & strong\,emphasis \end{Bmatrix}$$

This is a major improvement to existing optimisation algorithms, shifting drastically the results of the optimisation process.

With all modules of one possible solution path being mapped, they can be joint to a curve via connecting them, leading to a hyperspace curve. This procedure is performed for all determined possible paths, leading to a set of hyperspace curves for the individual configuration problem. These curves can be regarded as a set of polynomial functions of the n^{th} order in the interval *[0; Idealpoint S]*. These polynomial functions now form a set of twisted areas over the hypercube diagonal, which is exemplary shown for three dimensions $x_1 - x_3$ in the following Figure 2.

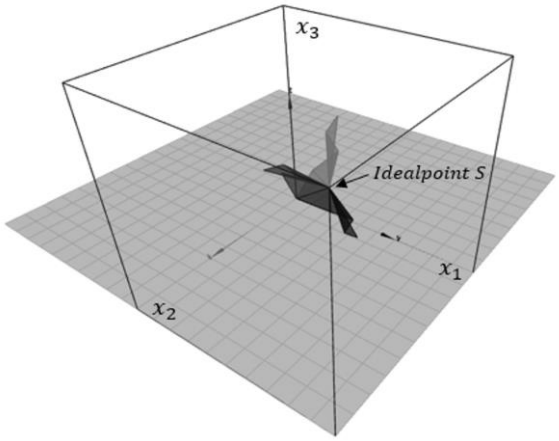

Figure 2. 3D-visualisation of hypercube with three exemplary paths in three dimensions x1 - x3

The hypercube diagonal ***d*** is thereby defined as the square root of the degree of dimensions ***n*** multiplied with the edge length ***a*** of the hypercube as seen in Equation (2):

$$d = \sqrt{n} * a \tag{2}$$

In order to determine the most balanced solution with respect to the customer's demands, the mapped path with the least spatial deviation to the ideally balanced space diagonal needs to be determined. Therefore, each dimensional area of the polynomial function over the hypercube diagonal is calculated.

At first, the individual functions need to be interpolated. There are several different interpolation methods to be found within the pertinent literature, from linear, partial, hermit, trigonometric to polynomial interpolation (Diestel, 2000).

With a set of *n + 1* data points (x_i, y_i) where no two are the same, the polynomial interpolation provides a suitable solution to this requirement (Equation 3).

$$P(x_i) = y_i \qquad for\,all\; i \in \{0,1,\ldots,n\} \tag{3}$$

This is done via a transformed Lagrange-Formula, leading to the use of the barycentric interpolation (Equation 4)

$$P(x) = \frac{\sum_{j=0}^{n} \frac{\lambda_j f_i}{x - x_j}}{\sum_{j=0}^{n} \frac{\lambda_j}{x - x_j}} \tag{4}$$

with the use of barycentric weights (Equation 5)

$$\lambda_j = \prod_{\substack{i=0 \\ j \neq 0}}^{n} \frac{1}{x_j - x_i} \tag{5}$$

These interpolated polynomial functions can then be used to calculate the spatial area of each dimension for each mapped path by integrating the polynomial function over the interval *[0; Idealpoint S]*. (Equation 6).

$$spatial\,area_i = \int_0^{Idealpoint\ S} P_{Dimension\ i}(x_i)\,dx \qquad with\, x_i \in \mathbb{D} \tag{6}$$

Having calculated the weighted spatial area of each dimension for one path, those areas are to be summed up, resulting in a dimensionless measure $deviation_i$ for the deviation of this individual product configuration from the hypothetic ideal solution. This measure is described within the following Equation (7).

$$deviation_i = \sum_{j=o}^{n} \int_{0}^{Idealpoint\ S} P_{Dimension\ i}(x_i)\,dx \qquad with\, i, j \in \mathbb{N} \tag{7}$$

Having determined the $deviation_i$ for all k_i *with* $k \in \mathbb{D}$, the most balanced product variant with respect to the weighted customer's demands as well as the available modular product architecture can be determined by deriving the minimal value of $deviation_i$, which is shown in the following Equation (8):

$$Configuration = min \sum_{k=0}^{n} \sum_{j=o}^{n} \int_{0}^{Idealpoint\ S} P_{Dimension\ i}(x_i)\,dx \qquad with\, i, k \in \mathbb{N} \tag{8}$$

Therefore, a multi-factor customer demand system can be balanced and different possible solutions weighted against each other.

Furthermore, the concept of regarding product variants as polynomial functions of the n^{th} order allows for the determination of the weakest modules within the modular architecture in a multi-dimensional way. By calculating the local and global maxima of each possible path (Equations 9 and 10) with respect to the relevant dimensions, an advanced statement for the enhancement of the existing modular architecture can be made. All modules on or close to the hyperspace point where

$$\frac{df(x_i)}{\mathrm{dx}} = 0 \tag{9}$$

and

$$\frac{d^2 f(x_i)}{dx} > 0 \tag{10}$$

are modules with a local or global maximum deviation from the ideal solution.

By constantly calculating and storing the information about those specific modules, a reasonable starting point for the enhancement of the existing modular product architecture is derived.

Another major improvement that comes with this algorithm is its flexibility an adaptability. As one requirement for the development of this algorithm is to integrate it within existing modular architecture data structures with their set of rules without making major changes to these structures, this algorithm comes with no limitations to the amount and type of influence factors, as they are all considered as normalized, dimension-less edges of an n-dimensional hypercube. Furthermore, even the current underlying modular architecture can be changed within one configuration process by parallelly adapting the MBSE-data structure and its set of dependencies. Especially with configure-to-order-based engineer-to-order process structures (Seiler et al., 2019b), which are usually found within special equipment manufacturers, this is an important issue. Adaptations to existing modules, which are already pre-designed during the modularisation phase with respect to the possibility of these adaptions, can be integrated into the proposed "modular road map" as a further crossing point, which can easily be blocked or accessed with changing the adjacency matrix values. This allows for fast and sustainable calculations of a single adaptions impact to the final product variant.

4. Validation: Case study

For validation purposes, the proposed algorithm was tested upon the modular product architecture of a german special equipment manufacturer (SME). This high-tech SME builds specially designed welding systems (batch size one) and already implemented a modular product architecture on the basis of the Integrated PKT-Approach.

Together with its set of intermodular constraints, this modular architecture forms a modular kit, which is currently mainly used within the sales product life phase. As no two offered or built systems are the same, the process flow can be seen as an ETO-based-CTO structure, with basic modules being adapted

or expanded according to the individual customers' demands. As there are different modules meeting the same customer relevant properties but with a differing characteristic, there clearly is the in this paper proposed issue about which module to choose for its possible adaption to the customer's needs. In total, within this SME 28 different modules with 108 interacting constraints can be identified, resulting in a high need of software support when configuring suitable product variants. For a better understanding of the in this contribution proposed algorithm, we reduced the case study modular architecture to eight modules with different characteristics, which meet in total five customer relevant properties. The set of constraints is furthermore limited to three interaction constraints.

The to be considered example for the validation case study is a simple welding task of a bent tube with two straight ends (customer product), which is to be welded along its mantel line by the to be configured SME-product. This SME-product is to be configured according to the customer's needs, which can implicitly be derived from the customer product's requirements. Due to the weld seam geometry of the bent tube, the welding system's performance and therefore its customer relevant properties can be determined. Figure 3 shows the exemplary paths acc. Figure 1 and 2 of this proposal.

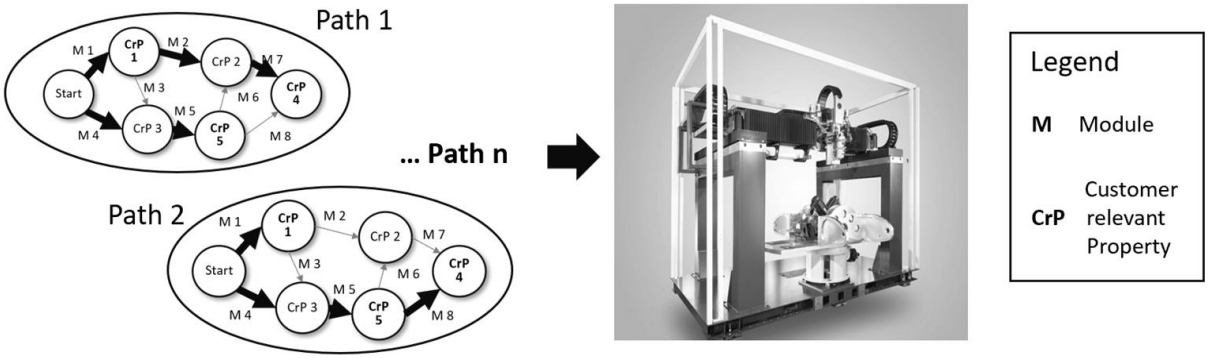

Figure 3. Variant paths for case study configuration determination

In the following, the generation of these paths and the derivation of the customer relevant properties on the basis of the individual customer's needs is described.

Let the modular kit of the SME, from which the modules for the final welding system are to be chosen, consist of the in Table 1 presented eight modules with three dimensions each. Those dimensions are material costs in €, delivery time in weeks and the estimated engineering amount in weeks for their adaption to the specific customer demands. The modules are on the one hand parts of a five-axis movement system for a NC-controlled tooling machine, and on the other hand modules for two different welding technologies (TIG-welding and laser welding).

The modules with the dimensions' characteristics are shown in the following Table 1.

Table 1. Case study modular kit with three dimensions and their expressions

Module #	Name	Dim 1 (Mat. cost)	D2 (Delivery time)	D3 (Engineering effort)
M1	X-Axis	10.000,00 €	10 weeks	2 weeks
M2	Y-Axis	10.000,00 €	10 weeks	2 weeks
M3	Z-Axis	80.000,00 €	10 weeks	2 weeks
M4	B-Axis	12.000,00 €	14 weeks	2 weeks
M5	A-Axis	14.000,00 €	14 weeks	2 weeks
M6	HF- power supply	25.000,00 €	3 weeks	3 weeks
M7	TIG-welding	54.000,00 €	3 weeks	0 weeks
M8	Laserwelding	80.000,00 €	8 weeks	0 weeks

Let there be three constraints which limit the use of the modular kit:

1. Welds must be performed in PA-position (position flat), therefore if Axis B is chosen, Axis A needs to be added as well
2. If the TIG-welding process is selected, a further linear movement axis (perpendicular to the main axis) for the ignition process needs to be chosen
3. The TIG-welding process requires the High-Frequency Power Supply module (HF-PS)

Furthermore, there are three customer relevant properties (CrP) to be met, resulting from the case study's welding task. First there needs to be a linear motion made possible for the welding of the flat parts, secondly a tilt-movement for the welding of the bent tube part and third a weld seam depth of 2 mm needs to be performed by the system.
Taking this modular kit with its constraints and the to be met customer relevant properties into account, the proposed configuration algorithm can be used to determine the most balanced solution with respect to different priorities being set for the three considered dimensions.
At first, using the modular roadmap and the underlying adjacency matrices, which can be seen in Figure 1 and Figure 2 of this contribution, all valid paths for this configuration problem can be determined. By the use of the adapted Dijkstra-algorithm, seven non-redundant paths can be isolated. Based upon the modules which are used to create every single path, the spatial areas of the twisted hyperspace polynomials can be calculated as described in section 3. In this case, we considered two different cases with different dimension priorities. Case 1 thereby describes a customer request with a strong emphasis on the total system's delivery time and slightly less respect to the overall system costs, whereas Case 2 has a strong emphasis on the system's cost part.
The following Table 2 shows the seven derived valid paths with their corresponding modules and the acc. to section 4 calculated spatial areas with respect to the different dimension priority emphasis.

Table 2. Case study valid paths with Case 1 and 2 calculated spatial areas

Valid paths	Modules	spatial area CASE 1 D1 = 1, D2 = 2	spatial area CASE 2 D1 = 2, D2 = 1
M1, M2, M4, M5, M7	X,Y, B, A, TIG	37,92	22,17
M1, M3, M4, M5, M8	X, Z, B, A, Laser	42,45	26,23
M1, M4, M5, M6, M7	X, B, A, HF-PS, TIG	33,77	20,54
M1, M4, M5, M8	X, B, A, Laser	34,12	***20,23***
M1, M2, M3, M5, M7	X, Y, Z, A, TIG	36,10	22,53
M1, M3, M5, M8	X, Z, A, Laser	32,30	20,60
M1, M3, M5, M6, M7	X, Z, A, HF-PS, TIG	***31,95***	20,91

As Table 2 shows, the varying prioritising of the product's dimension drastically changes the result for the as optimal derived modular solution. The encircled values within the column of Case 1 and 2 represent each the valid path with the overall minimal combined spatial area in all prioritised dimensions. With the emphasis mainly on the delivery time, the final SME-product is configured as a TIG-welding system with two linear axis and one rotational axis (Case 1). Case 2 instead focuses more on the cost side of the project, resulting in a completely different machine concept, as there is only one linear axis combined with two rotational axes and instead of the TIG-welding, laser welding is the chosen production process. After discussing these results with the respective experts of the case study SME, those results were stated as being valid. The following Figure 4 shows schematically the two different identified product variants. White modules thereby represent standard modules, which occur in every single product variant, the grey ones are variant modules, therefore modules with different characteristics which can be configured in order to meet individual customer relevant properties.

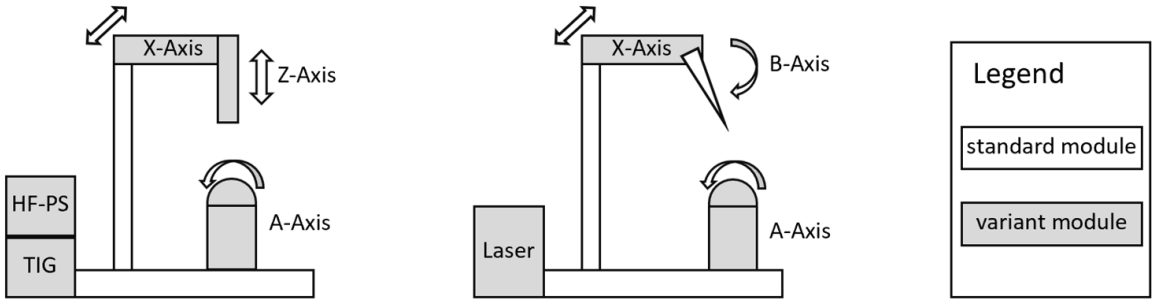

Figure 4. Schematic display of ideally balanced solutions for Case 1 (left) and Case 2 (right)

This result clearly shows the needs of a prioritised, multi-dimensional balancing algorithm for the proper determination of the most adequate solution to the independent customer's demands whilst considering all influencing factors and constraints of the modular architecture.

5. Conclusion and outlook

With the need for a multi-factor optimization problem being impertinent for a MBSE-based configuration software, the multi-dimensional hypercube offers a suitable solution when considering a large amount of influence factors. The determination of to be weighted product variants via the adapted Dijkstra-algorithm furthermore enables the consideration of only partial solutions, an issue being relevant especially when the business environment does not allow for a completely fixed modular architecture. When needing to react to individually, slightly changing customer demands, such as it can be observed within special equipment manufacturing, this use of partial solutions is key to choosing the most adequate product variant within the configuration process.

As the proposed algorithm considers the configuration problem not as a purely algebraic but as a geometric optimisation problem, it can cope with non-descript systems, as there is no need to calculate exact values but to minimize the total sum of twisted spatial areas by integrating dimension-wise and then adding up the integrated areas. Having determined the product variant with the least sum of spatial area integrals, the adequate product variant for the current customer demand is identified.

With the consideration of adaptions to the underlying modular architecture and therefore of a configure-to-order based engineer-to-order process structure, this algorithm proposes one step to determining the adequate product variant for the individual customer demand whilst still allowing for flexible adaptions within the final product variant. As all different influence factors as well was all dependencies within the underlying modular architecture still are cross-checked when implementing adaptions to a certain product variant, the risk of proposing contradictory solutions or offering the system with inappropriate pricing due to miscalculations, not taking all influence factors and dependencies into account, is minimized.

As a field for future research, the algorithms flexibility can be used for the analysis of different modular product architectures during their early design phases. As all modularisation methods have in common to not lead directly to only one possible solution of a modular architecture, until now, expert decisions and implicit expert knowledge have been used to determine the to be implemented modular architecture. This results in high cost and manpower efforts when noticing deficiencies of a modular architecture after its implementation. This algorithm might now be used to analyse different developed modular architectures before their implementation. By simulating customers' demands in a configuration software based upon this hypercube algorithm with those different modular architectures as a basis, a statement about the effectiveness of the different architectures can be made. Future research topics therefore are to develop a methodical measure about how to describe, measure and assess the effectiveness of modular architectures and then implement this measure in a hypercube-algorithm based configuration software.

References

Broumi, S. et al. (2019), "A new algorithm for finding minimum spanning trees with undirected neutrosophic graphs", *Granular Computing*, Vol. 4 No. 1, pp. 63-69.

Bursac, N. (2016), *Model Based Systems Engineering zur Unterstützung der Baukastenentwicklung imKontext der Frühen Phase der Produktgenerationsentwicklung, Dissertation*. Stolzenberger, Leimen.

Cicconi, P. et al. (2018), "A model-based simulation approach to support the product configuration and optimization of gas turbine ducts", In: *Computer-Aided Design & Applications*, Vol. 15, pp. 807-818. https://doi.org/10.1080/16864360.2018.1462564

Diestel, R. (2000), *Graphentheorie: Mathematical Physics and Mathematiks*. Springer, Heidelberg

Dijkstra, E.W. (1959), "A Note on Two Problems in Connexion with Graphs", In: *Numerische Mathematik* 1, pp. 269-271.

Erixon, G. (1998), "Modular Function Deployment: A Method for Product Modularisation", The Royal Insitute of Technology, Department of Manufacturing Systems, Stockholm.

Hadzic, T. and Andersen, H.R. (2006), *A BDD-Based Polytime Algorithm for Cost-Bounded Interactive Configuration*. In: *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*.

Hermann, M.-O., Michler, J. and Schönthaler, F. (2013), "Wo Kundenwünsche auf technische und wirtschaftliche Notwendigkeiten treffen" *Business News*, Vol. 3, pp. 13-16.

Krause, D. and Gebhardt, N. (2018), "Methodische Entwicklung modularer Produktfamilien: Hohe Produktvielfalt beherrschbar entwickeln", Springer, Hamburg. https://doi.org/10.1007/978-3-662-53040-5

Krieter, S. et al. (2018), "Propagating Configuration Decisions with modal Implication Graphs", In: *proceedings of 40th International Conference on Software Engineering. ICSE '18. ACM*, pp. 898-909. https://doi.org/10.1145/3180155.3180159

Liebisch, M. (2014), *Aspektorientierte Datenhaltung in Produktkonfiguratoren - Anforderungen, Konzepteund Realisierung. Dissertation*, Jena.

Seiler, F., Greve, E. and Krause, D. (2019a), "Development of a Configure-to-Order-Based Process for the Implementation of Modular Product Architectures: A Case Study". In *Proceedings of the 22nd International Conference on Engineering Design (ICED19)*, Delft, The Netherlands, 5-8 August 2019. https://doi.org/10.1017/dsi.2019.304

Seiler, F., Schwede, L.-N. and Krause, D. (2019b), MBSE-basierte Produktkonfiguratoren zur Analyse der Modularisierung bei der Entwicklung modularer Baukastensysteme. Entwerfen Entwickeln Erleben in Produktenwicklung und Design, Band 2, Dresden.

Simpson, T.W., Siddique, Z. and Jiao, R.J. (eds.) (2006), "Product platform and product family design: methods and applications." Springer Science & Business Media. https://doi.org/10.1007/0-387-29197-0_1

Stormer, H. (2007), "Kundenbasierte Produktkonfiguration", In *Informatik Spektrum*, Vol. 30. pp. 322-326. https://doi.org/10.1007/s00287-007-0177-1

Weisz, G., Györgi, A. and Szepesvári C. (2018), "LeapsAndBounds: A Method for approximately optimal algorithm configuration", *Proceedings of the 35th International Conference on Machine Learning*, Stockholm, Sweden, PMLR 80.

Zennaro, I. et al. (2019), "Big size highly customised product manufacturing systems: a literature review and future research agenda", In *International Journal of Production Research*, Vol. 57 No. 15-16, pp. 5362-5385. https://doi.org/10.1080/00207543.2019.1582819