3rd Conference on Production Systems and Logistics

Pareto Heuristic For Product Family-Oriented Product-Workstation Allocation Planning With Restricted Capacities

Christian Urnauer[1], Anna-Maria Larem[1], Joachim Metternich[1]
[1]*Institute of Production Management, Technology and Machine Tools, Darmstadt, Germany*

Abstract

A prerequisite for value stream design is the segmentation of products into families. This means that all products of a family are assigned to the same group of workstations so that the resulting material flows are separated as good as possible, and a higher degree of transparency is reached on the shop floor. However, since the number of workstations as well as their capacity is limited, shared resources cannot always be avoided in practice. Furthermore, the objective of product family orientation may compete with the objective of fulfilling product-workstation preferences. These preferences result, for example, from required equipment like specific tooling or from capability requirements. An optimization heuristic for this product-workstation allocation problem is presented within this article. First, the mathematical problem is formally described, then the heuristic is introduced and the required data for its application is outlined. For the evaluation, an extensive test set is generated, comparison heuristics are implemented, and solutions are made comparable through problem-specific bounds for both objectives. The results show that the solution quality of the pareto heuristic for both objective functions achieves almost the level of the comparison heuristics, which optimize only one of the objectives in isolation.

Keywords

value stream design; segmentation; pareto optimization

1. Introduction

Value stream design (VSD) is a widely used method to re-design material and information flows with the target of achieving a leaner, waste-reduced future state. The method is characterized by its simplicity and its application has led to significant improvements throughout different industries [1]. Practitioners particularly value the transparency and improvements achieved by the application of VSD [2]. However, in complex production environments, such as in high variety - low volume type companies, the application of the standard method is challenging [3]. Therefore, the increasing data availability in production can be facilitated to holistically support the value stream method with data analytics [4].

The segmentation of product families is a prerequisite for VSD. Families are identified by the similarity of the products' process steps in the downstream segment of the value stream, usually supported by a product-process-matrix [5]. In the segmentation, resources of each process are assigned to the product families with regards to their demand. This way, material flows are unbundled, the shop floor is structured, clear responsibilities can be assigned, departmental interfaces are reduced, and simpler planning and control mechanisms as well as the easier identification of improvements are facilitated through a higher degree of transparency [6]. However, experience from various VSD projects shows, that this step is frequently neglected and there are two major reasons for that:

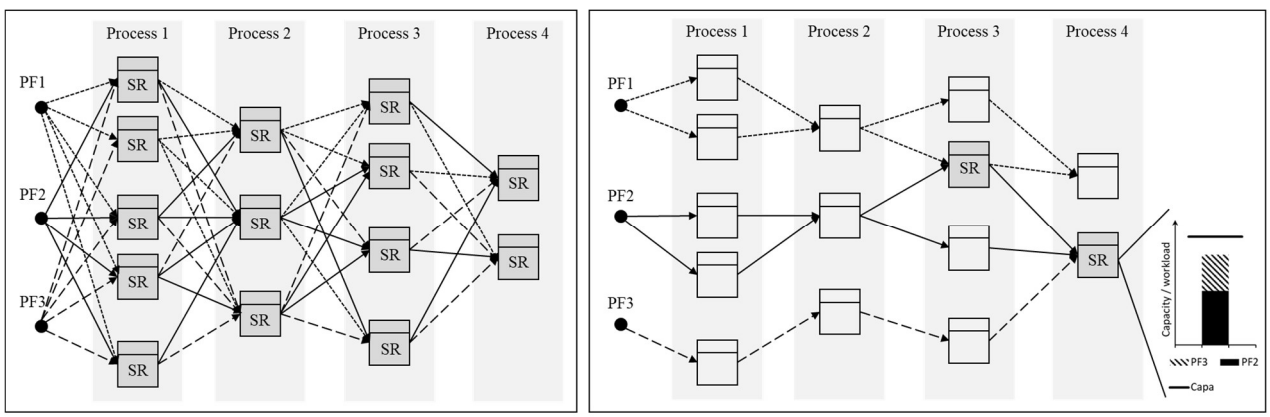
Firstly, identifying product families is particularly challenging if the product portfolio is very big and material flows are complex. This challenge can be countered with data analytics approaches applied to the product-process matrix [7]. Since the matrix can contain hundreds or thousands of products in mixed model value streams, a computer-aided sorting logic can help to visualize product family affiliations [8]. Furthermore, the application of cluster algorithms supports the identification of product families [4,7,9].

The second challenge in value stream segmentation is to realize a good product-workstation allocation that results in separable segments for the determined product families while considering workstation preferences and capacity restrictions. Following the standard design approach, product demand and workstation capacities are initially not taken into account and therefore, solutions are not practicable and must be adjusted iteratively [6]. Smaller adjustments can be achieved through optimizations in processing times, setup times, or availability. For bigger adjustments, operating times, shift models, the number of workers or machines must be increased or decreased, or products must be moved to other workstations with overcapacity. Depending on the initial situation, this typically results in a value stream design with shared resources and unevenly levelled workload. Additionally, workstation prioritizations are not taken into account in the standard approach. These prioritizations may result from restrictions such as required equipment, available workspace, degree of automation, or simply from personal preferences. In order to develop value stream oriented design proposals for such multi-dimensional solution spaces in complex production environments, smart optimization heuristics are required [10].

To support a product family-oriented segmentation under consideration of the available capacities, this paper proposes a heuristic planning approach. The multi-objective optimization problem as well as its restrictions are formulated in the next section, followed by the description of the product-workstation allocation planning heuristic and an outline of the required business data. Eventually, the approach is evaluated with a generated test set using assessment measures to estimate the solution quality in relation to the solution space and comparison heuristics that allow for contrasting with other feasible solutions.

2. Problem Definition

As illustrated in Figure 1, the product family-oriented segmentation transfers unclear material flows (left) to a clearly structured shop floor with fewer shared resources (right). This target is reached through a well-planned assignment of products to workstations which are limited in capacity. In the lower right of the figure, it is illustrated how the workload of two product families adds up on a shared resource but is covered by the slightly higher capacity of the respective workstation. Another aspect to consider in a well-planned product-workstation allocation is the workstation prioritizations that may exist for the products to be allocated.



PF = product family; SR = shared resource

Figure 1: Segmentation of the material flows of three product families

An ideal solution is characterized by:

A) an unambiguous *dedication of workstations to product families* so that clearly separatable segments are created with as few shared resources as possible, where
B) as many products as possible are *assigned to their most prioritized workstation.*

With the two independent objectives *product family separation* and *prioritization-based assignment*, this is a pareto optimization problem. It is characteristic for these kinds of problems that there is a set of pareto optimal solutions - the so called pareto front, for which no objective value can be improved without worsening the other. An optimal choice among these solutions depends on the decision maker's preferences.

The restrictions to this allocation problem are:

- every product belongs to one product family
- every product must be assigned to one workstation for every process it passes
- the total load of all assigned products must be covered by the capacity of the workstation

The mathematical problem formulation is outlined in the following. The objectives are described in section 0 and the restrictions are discussed in section 2.2. Since the product-workstation allocation for each process in the value stream is independent of the other processes' allocation, the problem is formulated for the consideration of one process. The used notations are summarized in Table 1.

Table 1: Notations

$FWA_{f,ws}$	Binary variable that indicates whether at least one product of family f is allocated to workstation ws
FWA_{total}	Total number of assignments of product families f to workstations ws
$Prio_{p,ws}$	Prioritization value for product p to be allocated to workstation ws
$Prio_{total}$	Total prioritization reached by all product-workstation assignments
$\emptyset\ Prio_{f,ws}$	Average prioritization value of all unassigned products of a family f for a workstation ws
$x_{p,ws}$	Binary decision variable that indicates whether product p is allocated to workstation ws
$z_{p,f}$	Binary parameter that indicates whether product p is part of product family f
y_p	Binary parameter that indicates whether product p is produced on this process
L_{ws}	Workload of workstation ws
C_{ws}	Capacity of workstation ws
$ct_{p,ws}$	Cycle time of product p at workstation ws
d_p	Demand per week for product p at the considered process
s_{ws}	Number of shifts for workstation ws
ot_{ws}	Operating time of workstation ws
OEE_{ws}	Overall equipment effectiveness for workstation ws
F	Set of product families
P	Set of products
WS	Set of workstations
f	Index for product families
p	Index for products
ws	Index for workstations

2.1 Objectives

$$Minimize\ FWA_{total} = \sum_{ws \in WS} \sum_{f \in F} FWA_{f,ws} \tag{1}$$

$$Maximize\ Prio_{total} = \sum_{ws \in WS} \sum_{p \in P} Prio_{p,ws} \cdot x_{p,ws} \tag{2}$$

with:

$$FWA_{f,ws} = \begin{cases} 1, & if \sum_{p \in P} x_{p,ws} \cdot z_{p,f} > 0 \\ 0, & else \end{cases} \qquad \forall\, ws \in WS, \forall\, f \in F \tag{3}$$

The binary decision variable of the problem is $x_{p,ws}$, which indicates whether product p is assigned to workstations ws ($x_{p,ws}$= 1) or not ($x_{p,ws}$= 0).

Equation (1) expresses objective A): the number of assignments of product families f to workstations ws must be minimized, so that resources are shared with as few product families as possible. These family-workstation assignments are described by the binary variable $FWA_{f,ws}$. In equation (3), $FWA_{f,ws}$ takes the value 1, if at least one item of product family f is assigned to workstation ws, and 0 otherwise. The binary parameter $z_{p,f}$ in this equation indicates whether a product p belongs to the product family f. Thus, the product of $x_{p,ws}$ and $z_{p,f}$ is 1 if a product of the considered family is assigned to the workstation ws via $x_{p,ws}$. Summing $FWA_{f,ws}$ for the set of all existing families F in equation (1) gives the number of families sharing a workstation ws. Summing this number of planned product families per workstation for the set of all workstations WS, results in the total number of family-workstation assignments to be minimized.

Equation (2) expresses objective B): assignments must fulfil the highest possible product-workstation prioritizations. $Prio_{p,ws}$ is a parameter indicating the preferences for each product p to be produced on workstation ws. In practice, these prioritizations can be collected in the form of a product-workstation prioritization matrix [7], in which $Prio_{p,ws}$ can take any real number. However, it makes sense to use a predefined scale for the value range when collecting the prioritizations. The product of $Prio_{p,ws}$ and $x_{p,ws}$ is summed for all products and workstations, so the result is a total prioritization value to be maximized.

2.2 Restrictions

$$\sum_{f \in F} z_{p,f} = 1 \qquad \forall\, p \in P \tag{4}$$

$$z_{p,f} \in [0,1] \qquad \forall\, p \in P, \forall\, f \in F \tag{5}$$

$$\sum_{ws \in WS} x_{p,ws} = y_p \qquad \forall\, p \in P \tag{6}$$

$$x_{p,ws} \in [0,1] \qquad \forall\, p \in P, \forall\, ws \in WS \tag{7}$$

$$y_p \in [0,1] \qquad \forall\, p \in P \tag{8}$$

$$L_{ws} \leq C_{ws} \qquad \forall\, ws \in WS \tag{9}$$

with:

$$L_{ws} = \sum_{p \in P} x_{p,ws} \cdot ct_{p,ws} \cdot d_p \qquad \forall\, ws \in WS \tag{10}$$

$$C_{ws} = s_{ws} \cdot ot_{ws} \cdot OEE_{ws} \qquad \forall\, ws \in WS \tag{11}$$

Equations 5, 7, and 8 define $z_{p,f}$, $x_{p,ws}$ and y_p to be binary. Since one product can only belong to one product family, the sum of $z_{p,f}$ over the set of all families F must be 1 for each product p, which is expressed by equation 4. If a product p passes the considered process during its production, the parameter y_p is 1.

Therefore, the sum of product-workstation assignments $x_{p,ws}$ over all workstations WS of this process must be equal to y_p (equation 6). This ensures that one workstation is assigned for each required process. Equation 9 is the capacity restriction. The total load of a workstation L_{ws} must always be less than or equal to its available capacity C_{ws}. The workload of each product is calculated as its demand at the considered process d_p multiplied with its cycle time at the considered workstation $ct_{p,ws}$. Equation 10 constructs the total load as the sum of all product assignments $x_{p,ws}$ multiplied with their cycle times and demands. Eventually, equation 11 gives the total capacity of a workstation as the product of its number of shifts s_{ws}, its available operating time ot_{ws}, and its overall equipment effectiveness OEE_{ws}. Since these are all external parameters, the capacity can be pre-calculated. However, since they are key levers for adjustments in the resulting segmentation, it makes sense to model them explicitly.

3. Heuristic For Product Family-Oriented Product-Workstation Allocation Planning

Since heuristics are typically more applicable to complicated conditions than exact methods, it is often useful to create an accurate representation of the real problem and solve it heuristically, rather than looking for an exact solution to a more abstracted model [11]. Since integer programming problems are known to be hard to solve within reasonable time, the use of a heuristic also makes sense regarding the computing time for large real-world instances. Furthermore, heuristics with good comprehensibility increase the acceptance for their solution especially for practical problems like the one introduced in this paper. In the following, a greedy pareto heuristic procedure for the product family-oriented product-workstation allocation planning problem with restricted capacities is introduced as well as an outline of the required input data.

3.1 Procedure

The procedure for the proposed greedy heuristic is described by the following pseudo-code:

```
1   DO UNTIL all products are assigned:
2       CALCULATE ∅ Prio_{f,ws} per unassigned product family and available workstation
3       SELECT product family-workstation combination with highest ∅ Prio_{f,ws}
4       FOR each prioritization level within selected combination in descending order:
5           FOR each product in descending order of required load ct_{p,ws} · d_p:
6               IF available capacity ≥ required load of product p:
7                   Assign product to workstation: x_{p,ws} = 1
8                   Reduce available capacity by allocated workload
9       Block product family-workstation combination for further iterations
10      IF workstation has not enough available capacity:
11          Remove ws from available workstations
12  END
```

The outlined procedure is iterated until all products are assigned to a workstation (line 1). First, the mean prioritization values $\emptyset\, Prio_{f,ws}$ are calculated for each product family-workstation combination with remaining unassigned products and available capacity (line 2). In line 3, the combination with the highest mean value is selected since it promises to involve many products with high prioritization values for the considered workstation. In the next step, the products of the selected product family are iteratively assigned to the selected workstation. Therefore, they are sorted in descending order of their prioritization value (line 4) and within each prioritization level, they are sorted in descending order of their required workload (line 5). Starting with the highest prioritization level, it is ensured that as many highly prioritized products as possible can be allocated to the selected workstation before its capacity may be exhausted. For each product it is checked if the available capacity of the considered workstation is still higher than its required load (line

6). If this is the case, the product is assigned to the workstation (line 7) and the remaining available capacity is reduced (line 8). If, on the other hand, the available capacity is not sufficient, the product remains unassigned and will be considered in further iterations. As soon as all products of the selected product family have been considered for the workstation assignment, the respective product family-workstation combination is blocked for all further iterations (line 9), since the workstation does not provide sufficient capacity for the remaining products of that family which are still to be assigned. Eventually, it is checked whether the selected workstation still has sufficient available capacity for further assignments in general (line 10). If this is not the case, it is removed from the list of available workstations. This procedure is repeated until all products have been assigned to a workstation.

Although the solution space may become very large for bigger problems, this heuristic is a simple and intuitive procedure to reach a good segmentation under consideration of allocation preferences and capacity restrictions. Its application is subject to some limitations regarding the capacity utilization. Firstly, it assumes a situation of scarce capacity so that in a case with large overcapacity it can result in a solution, where one workstation is assigned with two product families while another workstation is empty. In this case, the heuristic should be rerun with adjusted capacities. In the case of insufficient capacity, no feasible result is achieved since some products would remain unassigned. A splitting of the workload of one product does not take place. However, if splitting is necessary, it can be modelled by duplicating the corresponding product.

3.2 Required Input Data

For the application of the proposed heuristic, the following input data is required:

- Secondary demand for each product at each process d_p
- Cycle times per product and workstation $ct_{p,ws}$
- Product-process matrix with y_p
- Product-workstation-prioritization matrix with $Prio_{p,ws}$
- Product family assignment for each product $z_{p,f}$
- Workstation master data to calculate C_{ws}

The table with *secondary demand* d_p can be derived via the bill of material from the forecasted or historic overall demand of the finished products. *Cycle times* $ct_{p,ws}$ originate at best from time recordings or feedback data. However, planned times from the enterprise resource planning systems (ERP) can also be used with the knowledge in mind that they might be subject to inaccuracy. Multiplying d_p and $ct_{p,ws}$ gives the expected workload for each product at each workstation. A *product-process matrix* indicates which processes need to be passed by which product in the form of a binary matrix that holds the values for y_p. It is a standard tool for product family identification within the value stream method and if the information is not already available from the ERP, the matrix is either manually created or derived from transaction data [7]. The allocation preferences $Prio_{p,ws}$ are collected in the form of a *product-workstation prioritization matrix*. These are either assessed manually by experts or they can also be derived from historic production quantities mined from transaction data [7]. All products must be assigned to a *product family* which corresponds to the binary assignment parameter $z_{p,f}$. Eventually, the *workstation master data* must be given, including all information to calculate the available capacity C_{ws} of each considered workstation. These are the number of shifts s_{ws}, the operating time ot_{ws}, and the overall equipment effectiveness OEE_{ws}.

4. Evaluation

The proposed heuristic explores a new field in allocation planning. Thus, test data and assessment measures have not yet been established for a sound evaluation of the heuristic. Therefore, an extensive test set is

generated, assessment measures are introduced, and comparison heuristics are used to rank the solution quality regarding the underlying solution space. All evaluation calculations are implemented in Python 3.7.6 in Visual Studio Code and run on a MacBook Pro with 1.4 GHz Quad-Core Intel Core i5 and 8 GB RAM.

4.1 Generation of test set

The test set is inspired by a dataset from an industry project concerning the value stream design for a battery cell production, on which the heuristic has also been applied. The test set includes 7776 problem instances that are generated with the use of random numbers and a full-factorial design combining the parameters listed in Table 2. Since the workstation allocation problem for one process is independent from every other process in a value stream, each test instance only consists of *one process*. The *number of workstations* is varied within 3, 6, 9, and 12, and for each workstation in the test instance, 10, 25, or 50 times as many products are generated. Assignment *prioritizations* are allocated randomly, such that each product has a first choice with a high prioritization value ($Prio_{p,ws}$= 10) and a second choice with a lower prioritization value ($Prio_{p,ws}$= 2). For all other workstations the prioritization is 0. The *demand* for each product is either constant 1000 pieces for each product or uniformly distributed between 800 and 1200 pieces. *Cycle times* are randomly determined within the intervals [10, 190], [50, 150], [90, 110]. The cycle times are distributed in three different ways. They can be equal for one product on every workstation, which represents the easiest case for the product-workstation allocation with respect to the capacity alignment. They can also be equal for one workstation with every product, which represents an environment of very different technology levels on parallel resources, such as an automated assembly station next to a manual one. In the third case, all cycle times are set independently. Based on the demand and the average cycle times of each product, an expected total load can be calculated. The *capacity* level of the workstations in the test instances is determined as the total load divided by an average utilization factor. This way, the available workstation capacity $\emptyset C$ is set to a value which allows to expect an average *utilization* of 80%, 90%, 95%, or 98% through the allocated products. The actual utilization in the eventual solution can, however, differ due to the varying cycle times. Products are grouped into *product families* either randomly or with the use of cluster algorithms based on their workstation prioritizations, which imitates the process similarity that products within a family typically have. For this purpose, the ward algorithm with Euclidean distances and the average-linkage approach with cosine distances is used. The number of generated product families is set to 2, 4, or 6.

Table 2: Parameters for test instance generation

Symbol	Description	Values			
$\|WS\|$	Number of workstations	3	6	9	12
$\|P\|$	Number of products per workstation	10	25		50
-	Demand distribution	uniform		constant	
-	Cycle times distribution	by product	by workstation		independent
δ_{ct}	Spread of cycle times [time units]	[10, 190]	[50, 150]		[90, 110]
η	Average utilization of total capacity	80%	90%	95%	98%
$\|F\|$	Number of product families	2	4		6
-	Cluster method for product family formation	ward	average-linkage		random

4.2 Assessment of solution quality

The solution quality of the proposed pareto heuristic is assessed for the two objectives *total number of family-workstation assignments* FWA_{total} and *total prioritization* $Prio_{total}$ reached by the product-workstation assignments. To make the solutions of each problem instance comparable, FWA_{total} is set in relation with a lower bound LB and $Prio_{total}$ is set in relation with an upper bound UB. These bounds estimate the theoretically best possible values for the considered objective. As expressed in equation 12, the lower bound for FWA_{total} is either the number of product families or the number of workstations needed to cover the

entire workload. For the workload estimation, the average cycle time of each product $\emptyset ct_p$ is multiplied with its demand d_p and summed up for all products. The division of this workload by the specified workstation capacity $\emptyset C$ of the test instance results in the minimum number of required workstations. The upper bound for $Prio_{total}$ is the sum of the highest prioritization value of each product as described by equation 13.

$$LB = \max\{|F|, \left\lceil \frac{\sum_{p \in P} \emptyset ct_p \cdot d_p}{\emptyset C} \right\rceil\} \tag{12}$$

$$UB = \sum_{p \in P} Prio_{p,max} \tag{13}$$

Based on these bounds, the evaluation results are given as ratios:

$$FWA\ [\%] = \frac{FWA_{total}}{LB} \cdot 100\% \tag{14}$$

$$Prio\ [\%] = \frac{Prio_{total}}{UB} \cdot 100\% \tag{15}$$

Therefore, values > 100% result for FWA [%] and values between 0% and 100% for $Prio$ [%]. The bound based ratios make different problem instances comparable and give an estimation on the quality of a solution. However, it is depending on the problem's structure how close to 100% a feasible solution can get for the two objectives. To get a better idea about this, three greedy comparison heuristics are implemented:

- *Random*: All products are sorted in descending order by average load and randomly assigned to a workstation with free capacity.
- *Max Prioritization heuristic*: All products are sorted in descending order by average load and iteratively assigned to the workstation with the highest prioritization, provided its capacity still allows it.
- *Min FWA heuristic*: All products are separated by product family and sorted in descending order by average load; starting with the family that has the highest total load, products are assigned block wise to the next free workstation.

With these simple heuristics, feasible solutions can be created for each instance, indicating how good one objective value can get when ignoring the second objective. Furthermore, the randomly created solution indicates how far the product-workstation allocation of the pareto heuristic is from a bad solution. With these comparative values, a classification in the solution space can take place, enabling a well-founded evaluation of the results.

4.3 Results

All solutions were calculated by the pareto heuristic within less than 10 seconds. On average, its runtime was 2.3 seconds. 6847 instances can be assessed, 929 instances lead to an infeasible solution in at least one of the applied approaches, which is due to problem instances that are too restrictive in terms of capacity. 695 of these unsolved instances are generated with the highest utilization expectation of η = 0.98.

The total results for the tested heuristics on all evaluable instances are illustrated in Figure 2. Averaged over all instances, the *pareto heuristic* reaches a prioritization ratio of $Prio$ [%] = 62, which is 13 percentage points behind the result of the *max prioritization heuristic*. A "natural low end" seems to be 44 percentage points lower at a prioritization ratio of $Prio$ [%] = 17. An even smaller gap can be observed for the FWA ratio. Here, the *pareto heuristic* reaches FWA [%] = 119 on average, while the *min FWA heuristic* reaches FWA [%] = 114. Compared to the average objective value of FWA [%] = 348 of the randomly generated solutions, the *pareto heuristic* is particularly strong in the target of reducing shared resources for clearly separatable value stream segments.

The achievable FWA [%] and thus the achievable number of shared resources depends above all on the scarcity of capacity (see Table 3), since this makes the problem more restrictive.

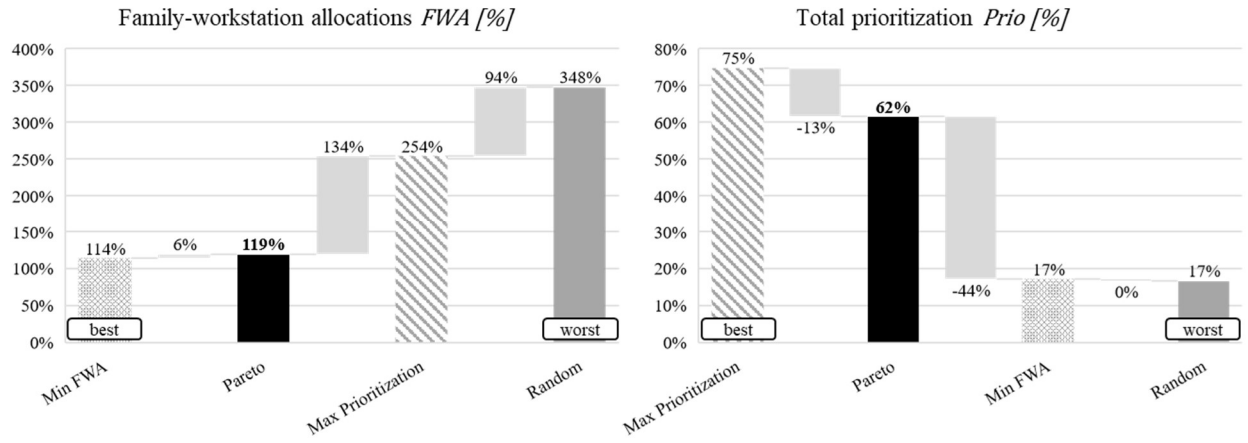


Figure 2: Average results for all heuristics in both objectives

If only those 4603 feasibly solved instances are considered for which the product families are formed using a clustering approach, the heuristic performs even better. In this case, the resulting values of the *pareto heuristic* are for both objectives almost as good as the comparison heuristic that ignores the other objective. The average FWA [%] is 117 compared to 113 of the *min FWA heuristic* and the average $Prio$ [%] is 73 compared to 75 of the *max prioritization heuristic*. Figure 3 illustrates the objective value pairs for these instances as well as their averages.

Table 3: Average FWA [%] for the pareto heuristic on all instances depending on the utilization factor η

η	FWA [%]
0.8	107.4
0.9	119.9
0.95	125.9
0.98	128.2

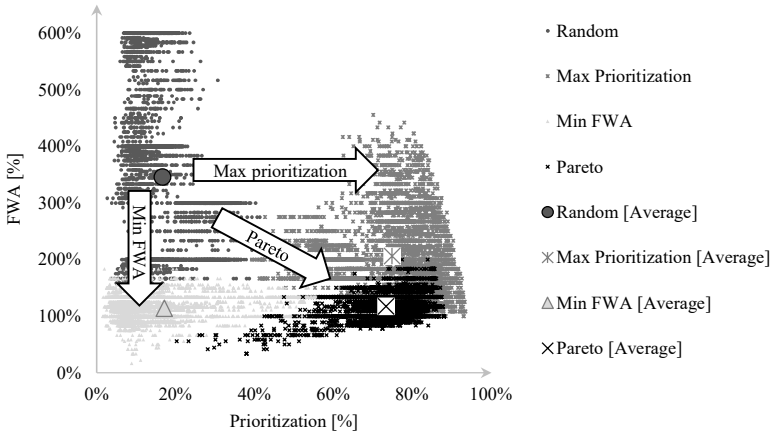


Figure 3: Results for all instances with clustering-based product families

5. Conclusion

This paper has introduced a pareto heuristic to support a product family-oriented segmentation under consideration of the available capacities as well as workstation preferences. The mathematical formulation of the multi-objective optimization problem with its restrictions is outlined, the heuristic procedure is presented, and the required business data is described. For the evaluation, an extensive test set is generated, assessment measures are introduced, and comparison heuristics are implemented. The evaluation shows, that the pareto heuristic performs strong for both objectives, especially for the target of reducing shared resources in a value stream design. As expected, even better results are achieved, if the product family affiliation correlates with the workstation preferences of the products. The value stream method provides to identify product families based on similarity in their required process steps and equipment. Therefore, high correlations between product family affiliation and workstation preferences should be observable in a well-

executed value stream design. The limitations of the heuristic are in its applicability for scenarios of insufficient capacity, large overcapacity, or if the high workload of one product would be required to be split. However, project-specific conditions can easily be modelled either via an adaption of the heuristic or via the given input data. The resulting product-workstation assignments of the pareto heuristic provides practitioners with a good and well-founded segmentation as a basis for the next steps in the value stream design that builds on it.

Acknowledgements

The research leading to these results has received funding from the German Federal Ministry of Education and Research under grant agreement number 02L19C150 (KompAKI).

References

[1] Serrano, I., Ochoa, C., Castro, R.D., 2008. Evaluation of value stream mapping in manufacturing system redesign. International Journal of Production Research 46 (16), 4409–4430.
[2] Lugert, A., Batz, A., Winkler, H., 2018. Empirical assessment of the future adequacy of value stream mapping in manufacturing industries. Journal of Manufacturing Technology Management 29 (5), 886–906.
[3] Braglia, M., Carmignani, G., Zammori, F., 2006. A new value stream mapping approach for complex production systems. International Journal of Production Research 44 (18-19), 3929–3952.
[4] Urnauer, C., Gräff, V., Tauchert, C., Metternich, J., 2020. Data-Assisted Value Stream Method. WGP Jahreskongress.
[5] Rother, M., Shook, J., 1999. Learning to see: Mapping the value stream to add value and eliminate waste. Lean Institute, Cambridge.
[6] Erlach, K., 2013. Value Stream Design. Springer, Berlin.
[7] Urnauer, C., Metternich, J., 2022. Product family identification based on data analytics. Procedia CIRP [in press].
[8] Duggan, K.J., 2013. Creating mixed model value streams: Practical lean techniques for building to demand, 2nd ed. CRC Press Taylor & Francis Group, Boca Raton, London, New York.
[9] Gaida, M., Günther, U., Wilsky, P., Riedel, R., 2020. Bildung von Produktfamilien als Planungsgrundlage auf Basis von Clusteralgorithmen. Zeitschrift für wirtschaftlichen Fabrikbetrieb 115 (3), 111–114.
[10] Erlach, K., Böhm, M., Gessert, S., Hartleif, S., Teriete, T., Ungern-Sterrnberg, R., 2021. Die zwei Wege der Wertstrommethode zur Digitalisierung: Datenwertstrom und WertstromDigital als Stoßrichtungen der Forschung für die digitalisierte Produktion. Zeitschrift für wirtschaftlichen Fabrikbetrieb 116 (12), 940–944.
[11] Reeves, C.M. (Ed.), 1993. Modern heuristic techniques for combinatorial problems, 1. publ ed. Halsted Press, New York.

Biography

Christian Urnauer, M. Sc. (*1990) is Research Associate at the Institute of Production Management, Technology and Machine Tools (PTW) at the Technical University of Darmstadt. He holds a Master of Science in Industrial Engineering which he obtained at the Technical University of Darmstadt. His research interests are in value stream mapping and data analytics.

Anna-Maria Larem, M. Sc. (*1994) is Manufacturing Trainee in the Graduate Specialist Program at Bosch Rexroth AG. She holds a Master of Science in Industrial Engineering which she obtained at the Technical University of Darmstadt. Her research interests are in lean production and data analytics.

Prof. Dr.-Ing. Joachim Metternich (*1968) is head of the Production Management division at the Institute of Production Management, Technology and Machine Tools (PTW) at the Technical University of Darmstadt. He is spokesman of the medium-sized businesses digital competence centre Darmstadt; he has been president of the International Association of Learning Factories (IALF) from 2015 to 2021 and he is a member of the scientific society for production technology (WGP) as well as scientific partner of the Conference on Production Systems and Logistics (CPSL).