Scalable co-Clustering using a Crossing Minimization – Application to Production Flow Analysis

Csaba Pigler, Ágnes Fogarassy-Vathy*

Department of Computer Science and Systems Technology,
University of Pannonia, Egyetem u. 10, H-8200 Veszprém, Hungary
piglercs@dcs.uni-pannon.hu, vathy@dcs.uni-pannon.hu

János Abonyi

Department of Process Engineering, University of Pannonia
Egyetem u. 10, H-8200 Veszprém, Hungary
Institute of Advanced Studies Kőszeg, Chernel u. 14, H-9730 Kőszeg, Hungary
janos@abonyilab.com

Abstract: Production flow analysis includes various families of components and groups of machines. Machine-part cell formation means the optimal design of manufacturing cells consisting of similar machines producing similar products from a similar set of components. Most of the algorithms reorders of the machine-part incidence matrix. We generalize this classical concept to handle more than two elements of the production process (e.g. machine - part - product - resource - operator). The application of this extended concept requires an efficient optimization algorithm for the simultaneous grouping these elements. For this purpose, we propose a novel co-clustering technique based on crossing minimization of layered bipartite graphs. The present method has been implemented as a MATLAB toolbox. The efficiency of the proposed approach and developed tools is demonstrated by realistic case studies. The log-linear scalability of the algorithm is proven theoretically and experimentally.

Keywords: cell formation; co-clustering; co-crossing minimization

[*] Corresponding author

1 Introduction

Industry 4.0 is focused on smart production in smart factories where there is direct communication between humans, machine, and resources. Smart products know their manufacturing process and future application [16, 25]. With this knowledge, they actively endorse the production process and the documentation ("when was I made, which parameters am I to be given, where I am supposed to be delivered."). Thanks to these developments that can be represented by 5C keywords (Connection, Cloud, Content, Community, and Customization) [17], complex production systems generate and store huge datasets, which has a high potential for productivity improvement. All these factors raise the question: "How can we increase productivity of manufacturing systems by the analysis of the huge amount of available data?"

There exists several potentials for Big Data, in manufacturing, such as, production management [24], supply chain planning [12], maintenance, and sales [8, 9]. Among these, production management is the most complex problem as it includes scheduling, optimization, (human) resource management, warehouse logistic, and manufacturing cell formation. Once this enormous amount of information is available from the components, the systems will be processed at the same time and the optimization of the manufacturing systems can significantly improve.

Cell formation (CF) is a widely studied topic in production systems optimization circles. The aim of cell formation is to create manufacturing cells from a given number of machines and products, by partitioning similar machines producing similar products. Since the formation of optimal manufacturing cells contributes significantly to the increased production, several different approaches have been proposed for the solution of a CF problems [5, 11, 15, 18, 23, 34]. Since, it can be difficult to find an optimal solution, in an acceptable amount of time, especially for large scale problems, usually heuristic approaches are used [19-22, 32, 33].

Furthermore, all these methods work only with relationships of machines and products (or with relationships of machines and parts). As these relationships can be represented by a two-layered bipartite graph or by an incidence matrix, the classical cell formation process can be considered as a biclustering task [1, 7].

Although, in complex manufacturing processes machines should be characterized by numerous properties, like the type of products, resources, and required skills from operators. To handle these elements of the production line, the traditional cell formation problem should be extended, and instead of the biclustering task, a co-clustering problem should be solved.

While biclustering algorithms can solve classical manufacturing CF process [1], we try to handle the extended multidimensional problem as a co-clustering task [10] based on crossing minimization of multipartite graphs. Since NP-hard problem, we utilize the widely applied heuristic barycentric method.

In a traditional cell formation problem, crossing minimization reorders the machines into cells, based on their similar part usage. As we mentioned before, cell formation problems can be more complex and they can require more properties to describe the whole production process. While dealing with these complex datasets, we developed a new crossing minimization method for multi-partite graphs. The proposed method sequentially reorders the elements of the node sets, thereby it relocates the elements of the connectivity matrix into a block-diagonal way. As result, the cell formation problem is handled in his original complexity.

Crossing minimization heuristics have been a subject of many years [4, 6, 13, 14, 28, 30, 31]. The complexity of these heuristics are linear or log-linear [26, 27]. This clearly indicates that the proposed problem formulation can lead to efficient solution for the multivariate cell formation problem.

Graphical representation of the manufacturing cell formation may also support optimization of production systems. Approaches like hierarchical clustering or Visual Assessment of Cluster Tendency (VAT) [2] are able to visualize the similarities of the elements, but by default they take only one or two variables into account. Nevertheless, complexities of these methods are not appropriate for Big Data [3], as the time complexity of VAT is $\mathcal{O}$ *(N^2)*, and the complexity of agglomerative clustering is $\mathcal{O}$ *(N^3)*, where *N* is the number of objects to be clustered.

In this article, we aim to present a novel production process optimization method which is able to accomodate more aspects (machines, suppliers, human resources, bill of materials, etc.) simultaneously and still be able to visualize the cell formation problem and it's solution accordingly for human interpretation. The optimization method is based on a newly developed co-crossing minimization method that solves the co-crossing minimization problem between $\mathcal{O}$ *(N)* and $\mathcal{O}$ *(N log N)* time, and therefore, able to process the Big Data rapidly and concurrently, the productivity of manufacturing systems can be significantly increased.

In the following we formulate the extended cell formation problem, then we present the novel co-crossing minimization algorithm that is able to handle this complex problem. The algorithm provides easy implementation and low computing complexity. Finally, we present the capability of our new method on different cell formation examples. Firstly, we compare our approach with the popular hybrid-heuristic cell formation algorithm [33] and show the applicability of the proposed performance measures. This will be followed by the numerical analysis of the scalability. Finally, an illustrative real-life example will be given.

2 Multidimensional Representation of the Cell Formation Problem

2.1 Classical Bipartite Graph-based Representation

Traditional cell formation problems are represented by bipartite graphs *(V,E)*, where *V* represents the set of vertices and *E* the set of edges. *V* is partitioned into two adjacent subsets. $V_0 = \{v_{0,1}, v_{0,2}, \dots, v_{0,N_0}\}$ represents the set of machines, and $V_1 = \{v_{1,1}, v_{1,2}, \dots, v_{1,N_1}\}$ the set of parts (see Figure 1

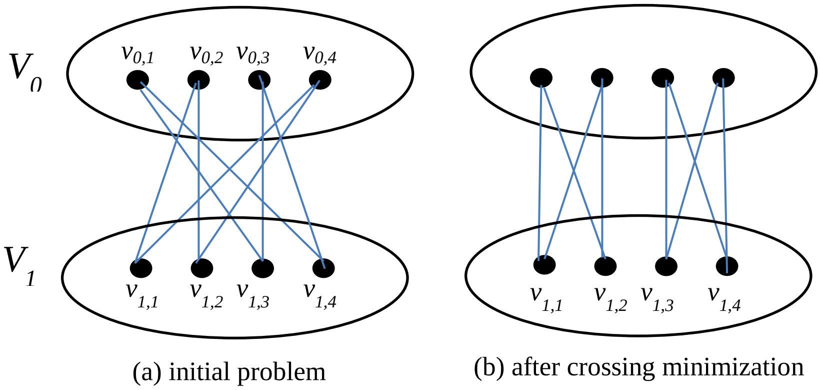


Figure 1

The classical cell formation problem is based on the crossing minimization of a bipartite graph, where V_0 represents the sets of machines and V_1 the set of the parts

The cell formation is based on the rearrangement of the order of the vertices. $\boldsymbol{o}_i$, $i = 0{,}1$, where $\boldsymbol{o}_i$ represents the sequence of all vertices of V_i . E.g. after the minimization of the crossings in the illustrative problem shown in Figure 1a, the sequence of the vertices becomes $\boldsymbol{o}_0 = [2{,}4{,}1{,}3]$**.**

The bipartite graph of the machine - part connections can also be represented by an interconnection matrix, $\boldsymbol{A}[\boldsymbol{o}_0,\boldsymbol{o}_1]$. The a_{ij} element of $\boldsymbol{A}[\boldsymbol{o}_0,\boldsymbol{o}_1]$ is 1 when the $\boldsymbol{o}_{0,i}$ -th machine uses the $\boldsymbol{o}_{1,j}$ -th part as input and otherwise 0. Please note, that the k-th element of these $\boldsymbol{o}_i$ vectors is the index which row (*i=0*) or column (*i=1*) is placed at the k-th place in the ordering. Later we are going to also use vector, $\boldsymbol{p}_{\mathrm{i}}$ to show the position of the vertices, $v_{i,j.}$ In our illustrative example as the first vertex, $v_{0,1,}$ is placed in the third place $p_{0,1}=3$, $\boldsymbol{p}_0 = [3{,}1{,}4{,}2]$**.**

From this viewpoint, the crossing minimization can be considered as reordering the rows and columns of the interconnection matrix to explore the hidden block-oriented structure of the matrix.

E.g. the initial problem is represented as: $$\boldsymbol{A}[\boldsymbol{o}_0,\boldsymbol{o}_1]=\begin{bmatrix}0&0&1&1\\1&1&0&0\\0&0&1&1\\1&1&0&0\end{bmatrix} \tag{1}$$

After crossing minimisation: $$\boldsymbol{A}[\boldsymbol{o}_0,\boldsymbol{o}_1]=\begin{bmatrix}\mathbf{1}&\mathbf{1}&0&0\\\mathbf{1}&\mathbf{1}&0&0\\0&0&\mathbf{1}&\mathbf{1}\\0&0&\mathbf{1}&\mathbf{1}\end{bmatrix} \tag{2}$$

As this simple illustrative example shows, minimization of the crossings provides not only a better visualization of the bipartite graph, but it also reorders the rows and columns of the incidence matrix in a block-diagonal way. With this order, similar nodes are placed next to each other, so they can form blocks that can be used to define manufacturing cells.

2.2 The Proposed Multidimensional Representation

In complex manufacturing processes, machines are characterized by numerous properties, like the type of products, resources, and the required skills of the operators. To handle all elements of the production line we extended the conventional cell formation task into a multidimensional problem. According to this goal, our key idea is to represent the cell formation problems by multi-layered graphs (see Figure 2).

The proposed *n*-dimensional representation is based on *n* sets of vertices

$$V=V_0\cup V_1\cup V_2\cup \ldots V_n,\ \ V_i\cap V_j=\emptyset\ ,\forall i\neq j \tag{3}$$

where each set represents possible values/categories of an attribute/feature of the machine. E.g. V_0 represents the set of machines, V_1 the parts, V_2 the products, V_3 the resources, V_4 skills of the operators.

Relationships between the machines and the i-th attribute of the production line can also be represented by a sparse matrix, $\boldsymbol{A}^{(i)}[\boldsymbol{o_0},\boldsymbol{o_i}]$, $i=1\ldots n$, where the dimensions of these matrices are $N_0\times\ N_i$.

The second fundamental idea is that the simultaneous re-ordering of the rows and columns of these matrices clusters the machines and supports the formulation and optimization of the manufacturing cells, as seen in Figure 2

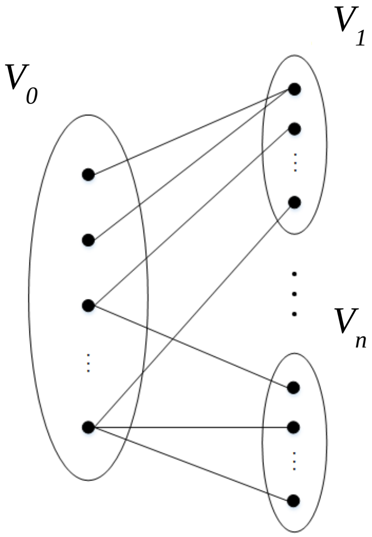


Figure 2

Representation of the multidimensional cell formation problem

The visualized benchmark cell formation problem has $N_0=20$ machines processing $N_1=34$ different parts, utilizing N_2=31 different resources, while working $N_3=37$ operators. The structure of this problem can be seen in the first row in Figure 3. The second row of this figure shows the rearranged data after the proposed co-crossing minimization algorithm which will be presented in the following section.

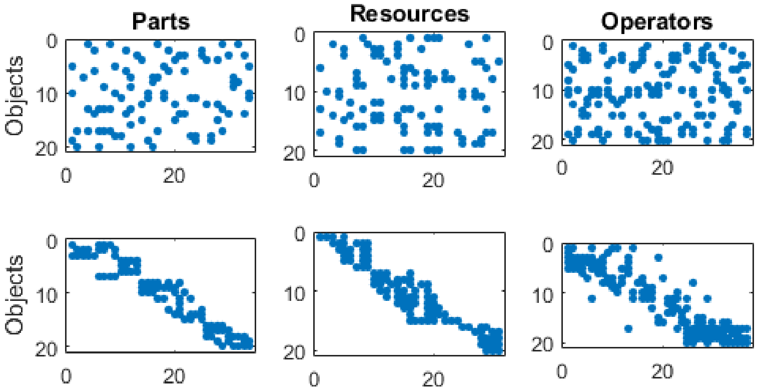


Figure 3

The benchmark cell-formation problem and the result of co-crossing minimization

3 Co-Crossing Minimization Algorithm

3.1 Barycentric Ordering of Crossing Minimization

The classical heuristic barycentric method iteratively reorders the $\boldsymbol{o}_0$ row and the $\boldsymbol{o}_1$ column orders of $\boldsymbol{A}[\boldsymbol{o}_0,\boldsymbol{o}_1]$ to reduce the number of the crossings. The reordering is based on the row and column barycenters of the $\boldsymbol{A}$ interconnection matrix. Barycenter heuristic assigns a new rank to each node based on the mean of ranks of its neighbor nodes.

The third key contribution of our paper is that we formulated the algorithm with the help of matrix operations to support compact, sparse matrix representation based implementation in of data analysis tools, like MATLAB.

The column barycenters are calculated as the mean of the places of the neighboring vertices

$$\boldsymbol{b}_i^C = \boldsymbol{p}_i^{\boldsymbol{T}}\boldsymbol{A}^{(\boldsymbol{i})}./\boldsymbol{s}_i^C \qquad (4)$$

where $\boldsymbol{p}_i$ represents the vector of the places of the vertices, $\boldsymbol{s}_i^C$ represents a row vector of the sum of the connections calculated as the sum of the columns of $\boldsymbol{A}^{(\boldsymbol{i})}$

$$\boldsymbol{s}_i^C = \boldsymbol{u}_0^R\boldsymbol{A}^{(i)} \qquad (5)$$

with the help of the $\boldsymbol{u}_0^R$, which is an N_0 sized unitary row vector.

With this formulation the j-th element of the $\boldsymbol{b}_i^C$ vector, $b_{i,j}^C$, can be interpreted as the average of the places of the machines that are connected to the j-th element of the i-th feature, $v_{i,j}$.

The row barycenters of $\boldsymbol{A}^{(\boldsymbol{i})}[\boldsymbol{o}_0,\boldsymbol{o}_i]$ are calculated similarly,

$$\boldsymbol{b}_i^R = \boldsymbol{A}^{(\boldsymbol{i})}\boldsymbol{p}_0\ ./\boldsymbol{s}_i^R \qquad (6)$$

where $\boldsymbol{s}_i^R$ represents the sum of rows of $\boldsymbol{A}^{(i)}$, calculated as

$$\boldsymbol{s}_i^C = \boldsymbol{A}^{(i)}\boldsymbol{u}_{\boldsymbol{i}}^{\boldsymbol{C}} \qquad (7)$$

where $\boldsymbol{u}_{\boldsymbol{i}}^{\boldsymbol{C}}$ is an N_i sized unitary column vector.

The standard crossing minimization algorithm iteratively reorders $\boldsymbol{o}_0$ based on shorting $\boldsymbol{b}_i^R$ and generates $\boldsymbol{A}[\boldsymbol{o}'_0,\boldsymbol{o}_1]$, than reorders $\boldsymbol{o}_1$ to generate $\boldsymbol{o}'_1$ based on the shorting the $\boldsymbol{b}_1^C$ barycenters of the columns of $\boldsymbol{A}[\boldsymbol{o}'_0,\boldsymbol{o}_1]$. The number crossings is minimised by repeating these orderings till convergence.

3.2 The Co-Crossing Minimization Algorithm

The key idea of the algorithm is that we simultaneously arrange the rows and the columns of the $\boldsymbol{A}^{(i)}[\boldsymbol{o}_0,\boldsymbol{o}_i]$ matrices. Since the row-orders of these matrices are identical, the row barycenters are calculated as the weighted sum of the row barycenters of the individual matrices:

$$\boldsymbol{b}^R = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{b}_i^R\, w_i = \sum_{i=1}^{n} \boldsymbol{A}^{(i)}\boldsymbol{p}_0\ ./\boldsymbol{s}_i^R\, w_i \tag{8}$$

The w_i weight can represent the importance of the features to their contribution to the cell-formation problem. When all features have equal importance, w_i should set as $w_i = 1/N_i$ to ensure that features with different number of categorical values have the same weight.

The steps of our new method can be seen in Algorithm 1.

```
**initialize the row orders**:
$\boldsymbol{o}_i, i = 0 \dots n$ as $1 : N_i$
**calculate the sum of the columns and rows**:
$\boldsymbol{s}_i^C, \boldsymbol{s}_i^R\ i = 1 \dots n$
**while** converge **do**
    calculate $\boldsymbol{b}_i^R, i = 1, \dots, n$, and $\boldsymbol{b}^{\boldsymbol{R}}$
    calculate the new $\boldsymbol{o}_0$ row order by shorting $\boldsymbol{b}^R$
    calculate the new $\boldsymbol{p}_0$ places of the vertices of $V_0$
    **for** $i = 1{:}n$ **do**
        calculate $\boldsymbol{b}_i^C, i = 1, \dots, n$
        calculate the new $\boldsymbol{o}_i$ row order by shorting $\boldsymbol{b}_i^C$
        calculate the new $\boldsymbol{p}_i$ places of the vertices of $V_i$
    **end for**
**end while**
```

Algorithm 1

The proposed co-crossing minimization algorithm

Please note that the $\boldsymbol{p}_i$ vectors are generated by sorting $\boldsymbol{o}_i$ the order vectors. The use of these vectors is important since the algorithm does not modify the original $\boldsymbol{A}^{(i)}$ space matrices, which ensures fast and memory effective implementation.

The algorithm stops when the ordering is converged or the maximum iteration number is reached.

3.3 Complexity Analysis

The complexity of the classical barycenter technique is $\mathcal{O}(|E|+|V|\ \ log\ V)$ [1,30]. It should be noted that since we decomposed the problem into n almost independent subproblems as $V = V_0 \cup V_1 \cup \dots V_n,\ V_i \cap V_j = \emptyset, \forall i \neq j$ and $E = E_1 \cup \dots E_n,\ E_i \cap E_j = \emptyset, \forall i \neq j$, the complexity of the algorithm is smaller than if we would handle the standard classical problem with the same size. By decreasing the sparsity of the problem (increasing the number of edges) the complexity linearly increases, while the increase of the number of vertices has log-linear effect as the critical step in the algorithm is, that it requires 2(*n+1)* sorts of the nodes in each iteration.

When this advantageous $\mathcal{O}$ *(N log N)* scaling of the quicksort algorithm is not enough, to achieve speed proportional to the size of the data $\mathcal{O}$*(N),* binsort can be applied. However, this requires auxiliary storage and memory for the bins. Because quicksort manipulates the data in place, it can sort larger arrays, albeit somewhat a bit slower. Another sorting option, is the application of the $\mathcal{O}$ *(N)* counting sort algorithm. It should be noted, that this and another advanced integer sorting algorithm requires the calculation of the median instead of the mean or rounding the ranks into integers. To scale the algorithm, it is possible to execute the loop iterations in parallel.

The algorithm solves the crossing minimization problem rapidly. Similarly to multi-layered application of crossing minimizatuion usually it stops after 5-10 iterations [1, 30]. It calculates the node ranks in each set of nodes linear algebraic way with matrix and vector multiplication. It should be noted, that matrices describing the cell formation problem are sparse, so they can be stored and handled efficiently.

Concluding, the proposed algorithm can be implemented in Big Data environments, as it supports sparse matrices, parallel computing (thanks to the barycenters of the interconnection matrices can be independently calculated), and the application of advanced (integer) sorting algorithms.

3.4 Performance Evaluation

The proposed method can be considered as a visualization tool, similarly to VAT (Visual Assessment of clustering Tendency) [2]. Although the resulted plots are informative, in most of the cases the numerical evaluation of the results is also necessary.

The first and most obvious measurement of the performance of the crossing minimization algorithms is based on the counting of the edge crossings. If we denote the rearranged $\boldsymbol{A}^{(i)}[\boldsymbol{o}_0,\boldsymbol{o}_i]$ matrix as $\boldsymbol{M}^{(i)}$, the number of crossings of the $v_{i,j}$ - th and $v_{i,k}$ - th vertices (represented by the j - th and k - th rows of the matrix) can be calculated as

$$nc^{(i)}(j,k) = \sum_{a=1}^{N_i-1} \sum_{b=a+1}^{N_i} m_{jb}^{(i)}\, m_{ka}^{(i)} \tag{9}$$

The total number of crossings of $\boldsymbol{M}^{(i)}$ can be calculated based on the sum of these $nc^{(i)}(j,k)$ values:

$$nc^{(i)} = \sum_{j=1}^{N_0-1} \sum_{k=j+1}^{N_0} nc^{(i)}(j,k) \tag{10}$$

Since the crossings of the E_i and E_j $\forall i \neq j$ edges are not taken into account, the total number of crossings is calculated as

$$nc = \sum_{i=1}^{n} nc^{(i)}$$

The effectiveness of the ordering from the expected block-diagonality of the resulted $\boldsymbol{M}^{(i)}$ matrices can be measured based on the distance of the nonnegative elements from the diagonal

$$d_{j,k}^{(i)} = m_{jk}^{(i)} \left| \frac{j}{N_0} - \frac{k}{N_i} \right|.$$

The percentage of the non-negative neighbors is also informative to represent the coherence of the resulted "maps", $nn_{j,k}^{(i)}$. Based on the combination of these two measures the we propose the following "goal-oriented" measure of the $\boldsymbol{M}^{(i)}$ ordering:

$$q^{(i)} = \sum_{j=1}^{N_0} \sum_{k=1}^{N_1} \frac{\left(1 - nn_{j,k}^{(i)}\right) d_{j,k}^{(i)}}{m_{j,k}^{(i)}}$$

It is also important to note, that the smaller the q value the better the ordering is in the data matrix.

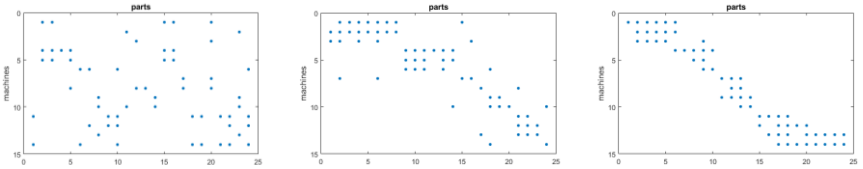
4 Case Studies

Manufacturing cell formation is widely studied and well-documented problem of process flow analysis. In this session, we provide several reproducible comparisons based on the most widely applied benchmark problems. The MATLAB implementation of the algorithm and the related datasets are downloadable from the website of the authors (www.abonyilab.com).

Firstly, we compare our approach with the popular hybrid-heuristic cell formation algorithm [33] and show the applicability of the proposed performance measures. This will be followed by the numerical analysis of the scalability. Finally, an illustrative real-life example will be given.

4.1 Application on Benchmark Problems

Since there are several two-dimensional examples for manufacturing cell formation problems in the literature, we first applied our method on one of those available [33]. The chosen dataset consists of 14 machines and 24 parts. The sparsity pattern of the incidence matrix is depicted in Figure 4(a). The hybrid heuristic algorithm is one of the recently published methods which combines the simulated annealing with genetic methods. This method in many cases outperforms the performance of the classical methods [33]. Solutions provided by the hybrid heuristic algorithm and our method can be seen in Figure 4. As it can be seen, our new approach provides much better block-diagonal ordered solution.



(a) Original dataset

(b) Result of hybrid heuristic method

(c) Result of the proposed method

Figure 4
The original data set and two orderings provided by the hybrid heuristic and our new crossing minimization algorithm

Using the number of edge crossings the hybrid heuristic and the proposed method can be numerically compared. While the original dataset has 674 edge crossings, and the result of the hybrid heuristic method has 256, the proposed crossing minimization algorithm provided an ordering only with 95 edge crossings.

We also compared crossing numbers of another eight benchmark datasets. The results are showed in Figure 5a, where the blue bars show the original, the red bars the hybrid heuristic, and the green ones the number of the crossings of the proposed algorithm. As it can be seen our method outperforms the hybrid heuristic method in every benchmark examples.

A summary of the previously presented *cn* crossing number and *q* block-diagonal ordering results can be seen in Table 1. As this table and Figure 5b show, the crossing minimization also ensures effective block-oriented orderings of these benchmark problems.

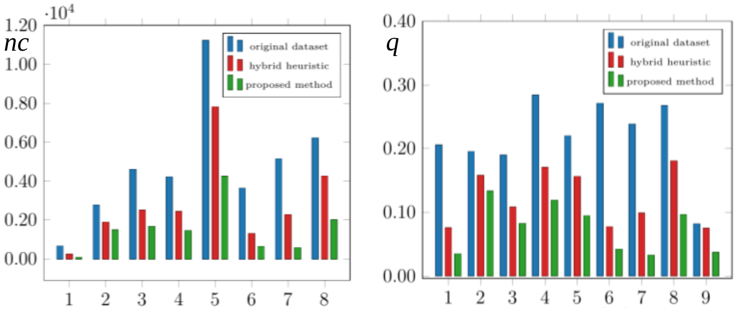


Figure 5
The number of the edge crossings (a) and the measure of the block-diagonal (b) ordering of the benchmark datasets

Table 1

Block-diagonal ordering (*q*) and crossing number (nc) results on different datasets

Dataset	Original		Hybrid heuristic		Proposed method	
	q	*cn*	*q*	*cn*	*q*	*cn*
P1(61)	0.2057	674	0.0756	256	0.0351	95
P2(62)	0.1954	2785	0.1580	1900	0.1336	1506
P3(63)	0.1900	4612	0.1083	2508	0.0826	1688
P4(64)	0.2840	4221	0.1708	2443	0.1193	1467
P5(65)	0.2201	11224	0.1564	7810	0.0948	4258
P6(66)	0.2713	3644	0.0775	1313	0.0422	653
P7(67)	0.2388	5150	0.0991	2283	0.0328	585
P8(68)	0.2675	6221	0.1811	4267	0.0966	2019
P9(69)	0.0786	229570	0.0782	173961	0.0373	131472

4.2 Crossing Minimization of Multidimensional Datasets – Numerical Analysis of the Complexity

While there are only 2D examples in the literature, we have generated several multidimensional benchmark problems to test the presented algorithm. After applying the previously mentioned iterative co-crossing minimization algorithm on the given multidimensional cell formation problem the results are seen in Figure 3, in the second row. As results, each matrix is reordered in a block-diagonal way.

In the following, we validate the log-linear scalability of the algorithm. We defined problems with 10, 100 and 500 properties/categories (N_i) of i=*1-6* features. Figure 6 shows the computational costs of these cases as we increased the number of objects (N_0) from 10 to 1.000.000. These graphs on the log-log scale show the log-linear complexity of the algorithm.

Figures 6-8 present similar results where the effects of increasing the number of machines, categories and properties are shown.

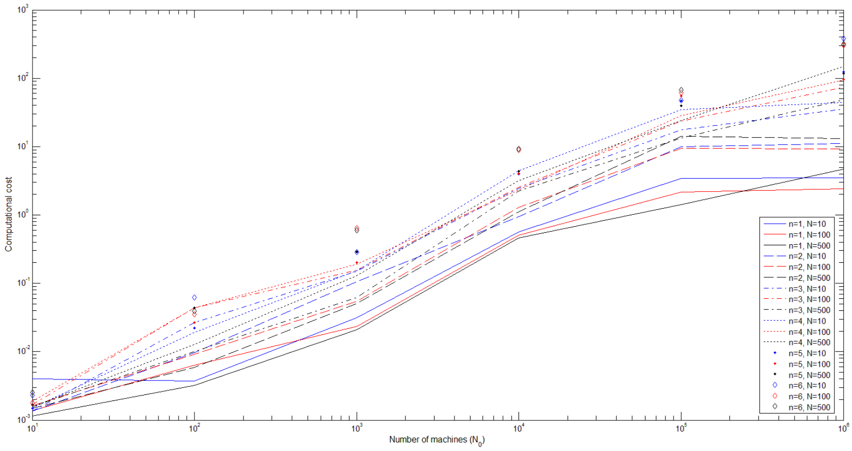


Figure 6

The computational costs of the proposed co-crossing minimization algorithm on different datasets

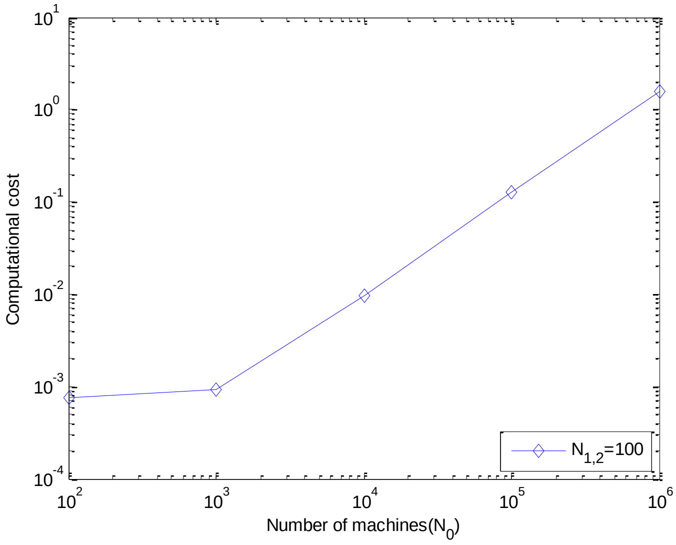


Figure 7

The increase of the number of machines linearly increases the computational complexity

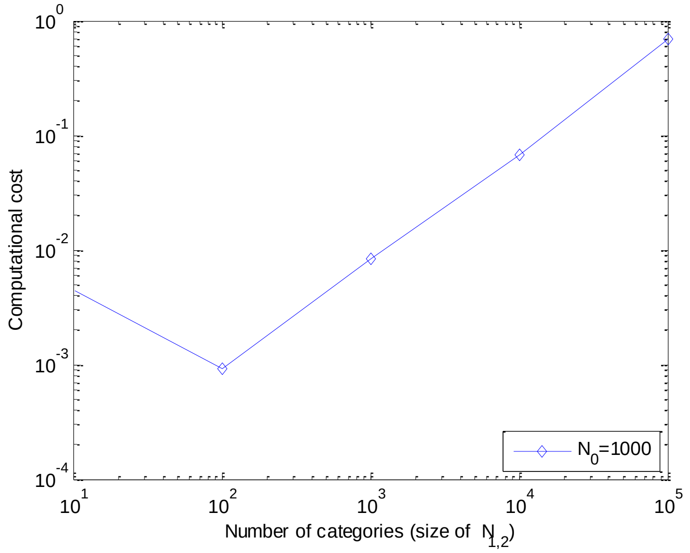


Figure 8

The increase of the number of categories linearly increases the computational complexity

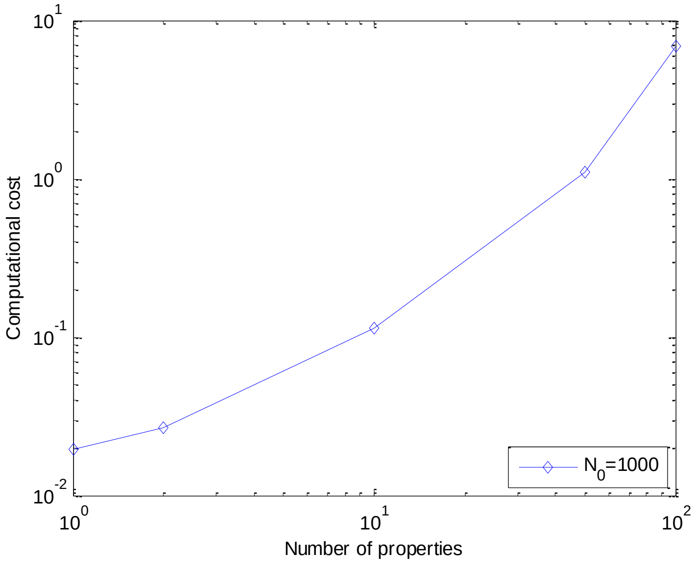


Figure 9

Effect of the increase of the number of properties (*n*)

As we will see, the results confirm the theoretical considerations and the industrial-scale applicability of the method.

4.3 Application on Real Life Problem

The previously presented multivariate co-crossing minimization method was applied on a real life example as well. We used this method on a production line analysis problem, where the primary task is the optimization of the production line of that produces over 5000 types of products assembled from several parts. Assuming that the switching time between similar products is less than the switching time between more different products, our aim was to analyze interconnections between parts and products. Since there are several stages of the production and different types of parts are used in different stages we formulated the multivariate model based on the hierarchy of the bill of materials (BOM) [29]. As Fig. 10 illustrates the methodology worked perfectly, we were able to sort the products according to their similarity.

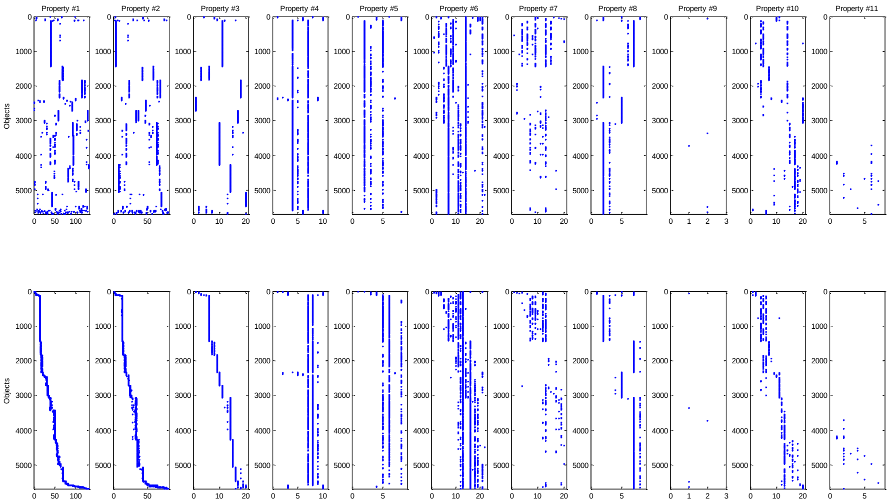


Figure 10

Real-life example for shorting products based on bill of materials

Conclusions

Thanks to the fourth industrial revolution production processes are becoming more and more integrated. This integration allows the simultaneous optimization of the whole supply chain. From this viewpoint production flow analysis is becoming an important tool since the analysis of the integrated marketing, design, production, logistic and sales data can effectively support production scheduling and flexible manufacturing cell formation.

Complex and integrated manufacturing processes require more detailed problem representation than used in classical manufacturing cell formation. This means, production lines should be characterized by several features that should be simultaneously analysed. We proposed a novel multipartite graph based representation of these complex production flow analysis problems and proposed an efficiently scalable clustering and visualization algorithm that sequentially reorders the edge crossings of bipartite graphs. The barycentric heuristic based co-crossing minimization method is simultaneously reorders the rows and columns of the interconnection matrices of the features to highlight their hidden block-diagonal structure, which structure supports data visualization easier. The applicability of the proposed co-crossing minimization is illustrated by several case studies.

We also showed that the proposed algorithm requires low computational capacities as it uses simple linear algebraic operations defined on sparse matrices. Another advantage of the method is its capability of parallelization and handling multi-dimensional sparse datasets. Considering these benefits, the proposed co-crossing minimization method can be scaled and used efficiently when we are dealing with large databases.

Acknowledgement

This publication/research has been supported by the National Research, Development and Innovation Office – NKFIH, through the project OTKA – 116674 (Process mining and deep learning in the natural sciences and process development).

Notations

Representation	
G	graph
V	vertex set of a G graph
V_i	set of vertices, set of objects/properties
$\mathrm{v}_{\mathrm{i,j}}$	j-th vertex (node) of the i-th set of vertices
N	number of nodes in a network/graph
N_i	number of nodes in the i-th set of vertices
E	edge set of a G graph
e_{ij}	edge between node i and j
$\boldsymbol{A}^{(i)}$	incidence (interconnection) matrix
$\boldsymbol{A}^{(i)}[\boldsymbol{o}_0, \boldsymbol{o}_i]$	ordered interconnection matrix
$\boldsymbol{o}_i$	ordering of the i-th vertex set
$\boldsymbol{p}_i$	positions of the vertices according to the $\boldsymbol{o}_i$ ordering
Crossing minimization	
$\boldsymbol{b}_i^C, \boldsymbol{b}_i^R$	column and row barycenters (vectors)
$\boldsymbol{s}_i^C, \boldsymbol{s}_i^R$	sum of the coumns and rows of $\boldsymbol{A}^{(i}$
Metrics	
$nc^{(i)}(j,k)$	number of crossings of the $v_{i,j}$ - th and $v_{i,k}$ - th vertices
$d_{j,k}^{(i)}$	diagonal distance
$q^{(i)}$	block-oriented quality of the ordering of the i-th intterconneciton matrix

References

[1] Ahmad, W., & Khokhar, A., cHawk : An Efficient Biclustering Algorithm based on Bipartite Graph Crossing Minimization. Computer Engineering, (May 2008) 249-263, http://doi.org/10.1021/es0602492, 2007

[2] Bezdek, J. C., Hathaway, R. J.: Vat: A Tool for Visual Assessment of (Cluster) Tendency, International Joint Conference on Neural Networks IJCNN'02, Vol. 3, pp. 2225-2230, 2002

[3] Chen, C. L. P., Zhang, C.-Y.: Data-Intensive Applications, Challenges, Techniques and Technologies: A Survey on Big Data, Information Sciences Vol. 275, pp. 314-347, 2014

[4] Chimani, M., Mutzel, P., & Bomze, I., A New Approach to Exact Crossing Minimization. Proc. of European Symposium on Algorithms (ESA~2008), 284-296, http://doi.org/10.1007/978-3-540-87744-8_24, 2008

[5] Dimopoulos, C., Mort, N.: A Hierarchical Clustering Methodology based on Genetic Programming for the Solution of Simple Cell-Formation Problems. International Journal of Production Research, Vol. 39, pp. 1-19, 2001

[6] Eiglsperger, M., Siebenhaller, M., & Kaufmann, M., An Efficient Implementation of Sugiyama’s Algorithm for Layered Graph Drawing. Journal of Graph Algorithms and Applications, 9(3), 305-325, http://doi.org/10.7155/jgaa.00111, 2005

[7] Erten, C., & Sözdinler, M., A Robust Biclustering Method Based on Crossing Minimization in Bipartite Graphs. 16th International Symposium on Graph Drawing, 5417, 439-440-440, http://doi.org/10.1007/978-3-642-00219-9_45, 2009

[8] Fan, C.-Y., Fan, P.-S., Chan, T.-Y., & Chang, S.-H., Using Hybrid Data Mining and Machine Learning Clustering Analysis to Predict the Turnover Rate for Technology Professionals. Expert Systems with Applications, 39(10), 8844-8851. http://doi.org/10.1016/j.eswa.2012.02.005, 2012

[9] Gansner, E. R., Koutsofios, E., North, S. C., & Vo, K. P. a V. K. P., A Technique for Drawing Directed Graphs- A Technique for Drawing Directed Graphs. Software Engineering, IEEE Transactions on, 19(3), 214-230, http://doi.org/10.1109/32.221135, 1993

[10] Gao, B., Liu, T.-Y., Zheng, X., Cheng, Q.-S., Ma, W.-Y.: Consistent Bipartite Graph Co-Partitioning for Star-structured High-Order Heterogeneous Data Co-Clustering, Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, 2005

[11] Goncalves, J. F., Resende, M. G. C.: An Evolutionary Algorithm for Manufacturing Cell Formation. Computers and Industrial Engineering, Vol. 47, pp. 247-273, 2004

[12] Hazena, B. T., Booneb, C. A., Ezellc, J. D., Jones-Farmerc, L. A.: Data Quality for Data Science, Predictive Analytics, and Big Data in Supply Chain Management: An Introduction to the Problem and Suggestions for Research and Applications, International Journal of Production Economics Vol. 154, pp. 72-80, 2014

[13] Jünger, M., & Mutzel, P. 2-Layer Straightline Crossing Minimization: Performance of Exact and Heuristic Algorithms. Journal of Graph Algorithms and Applications, 1(1), 1-25, http://doi.org/10.7155/jgaa.00001, 1997

[14] Khan, S., Bilal, M., Sharif, M., & Khan, F. A., A Solution to Bipartite Drawing Problem using Genetic Algorithm. Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6728 LNCS(PART 1), 530-538, http://doi.org/10.1007/978-3-642-21515-5_63, 2011

[15] Kusiak, A., Cho, M.: Similarity Coefficient Algorithms for Solving the Group Technology Problem, International Journal of Production Research Vol. 30, pp. 2633-2646, 2007

[16] Lee, J., Kao, H.-A., Yang, S.: Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment, Procedia CIRP Vol. 16, pp. 3-8, 2014

[17] Lee, J., Lapira, E., Bagheri, B., & Kao, H., Recent Advances and Trends in Predictive Manufacturing Systems in Big Data Environment. Manufacturing Letters, 1(1), 38-41, http://doi.org/10.1016/j.mfglet.2013.09.005, 2013

[18] Li, S., & Mehrabadi, H., Generation of Block Diagonal forms Using Hierarchical Clustering for Cell Formation Problems. Procedia CIRP, 17, 44-49, http://doi.org/10.1016/j.procir.2014.01.143, 2014

[19] Mahdavi, I., Teymourian, E., Baher, N. T., & Kayvanfar, V., An Integrated Model for Solving Cell Formation and Cell Layout Problem Simultaneously Considering New Situations. Journal of Manufacturing Systems, 32(4), 655-663, http://doi.org/10.1016/j.jmsy.2013.02.003, 2013

[20] Naadimuthu, G., Gultom, P., & Lee, E. S., Fuzzy Clustering in Cell Formation with Multiple Attributes. Computers and Mathematics with Applications, 59(9), 3137-3147, http://doi.org/10.1016/j.camwa.2010.02.038, 2010

[21] Oliveira, S., Ribeiro, J. F. F., & Seok, S. C., A Comparative Study of Similarity Measures for Manufacturing Cell Formation. Journal of

Manufacturing Systems, 27(1), 19-25, http://doi.org/10.1016/j.jmsy.2008.07.002, 2008

[22] Oliveira, S., Ribeiro, J. F. F., & Seok, S. C., A Spectral Clustering Algorithm for Manufacturing Cell Formation. Computers & Industrial Engineering, 57(3), 1008-1014, http://doi.org/10.1016/j.cie.2009.04.008, 2009

[23] Onwubulo, G. C., Mutingi, M.: A Genetic Algorithm Approach to Cellular Manufacturing Systems. Computers and Industrial Engineering, Vol. 39, pp. 125-144, 2001

[24] Sagiroglu, S., Sinanc, D.: Big Data: A review, International Conference on Collaboration Technologies and Systems, pp. 42-47, 2013

[25] Schuh, G., Potente, T., Wesch-Potente, C., Weber, A. J., Prote, J.-P.: Collaboration Mechanisms to increase Productivity in the Context of Industrie 4.0, ScienceDirect Procedia CIRP Vol. 19, pp. 51-56, 2014

[26] Shahrokhi, F., Sýkora, O., Székely, L. A., & Vrťo, I., On Bipartite Drawings and the Linear Arrangement Problem. SIAM Journal on Computing, 30(6), 1773, http://doi.org/10.1137/S0097539797331671, 2001

[27] Shiyas, C. R., & Madhusudanan Pillai, V., A Mathematical Programming Model for Manufacturing Cell Formation to Develop Multiple Configurations. Journal of Manufacturing Systems, 33(1), 149-158, http://doi.org/10.1016/j.jmsy.2013.10.002, 2014

[28] Stallmann, M., Brglez, F., & Ghosh, D., Heuristics, Experimental Subjects, and Treatment Evaluation in Bigraph Crossing Minimization. Journal of Experimental Algorithmics, 6(212), 8–es, http://doi.org/10.1145/945394.945402, 2001

[29] Stonebraker, P. W.: Restructuring the Bill of Material for Productivity: A Strategic Evaluation of Product Configuration, International Journal of Production Economics Vol. 45, pp. 251-260, 1996

[30] Sugiyama, K., Tagawa, S., & Toda, M., Methods for Visual Understanding of Hierarchical System Structures. IEEE Transactions on Systems, Man, and Cybernetics, 11(2), 109-125, http://doi.org/10.1109/TSMC.1981.4308636, 1981

[31] Warfield, J. N., Crossing Theory and Hierarchy Mapping. IEEE Transactions on Systems, Man, and Cybernetics, 7(7), 505-523, http://doi.org/10.1109/TSMC.1977.4309760, 1977

[32] Wu, T.-H., Chang, C.-C., & Chung, S.-H., A Simulated Annealing Algorithm for Manufacturing Cell Formation Problems. Expert Systems with Applications, 34, 1609-1617, http://doi.org/10.1016/j.eswa.2007.01.012, 2008

[33] Wu, T.-H., Chang, C.-C., & Yeh, J.-Y., A Hybrid Heuristic Algorithm Adopting both Boltzmann Function and Mutation Operator for Manufacturing Cell Formation Problems. International Journal of Production Economics, 120(2), 669-688, http://doi.org/10.1016/j.ijpe.2009.04.015, 2009

[34] Wu, T.-H., Low, C., Wu, W.-T.: A Tabu Search Approach to the Cell Formation Problem. International Journal of Advanced Manufacturing Technology Vol. 23, pp. 916-924, 2004