Comput Manag Sci (2012) 9:323–338
DOI 10.1007/s10287-012-0146-2

ORIGINAL PAPER

An exact model for cell formation in group technology

Dmitry Krushinsky · Boris Goldengorin

Received: 14 September 2011 / Accepted: 25 April 2012 / Published online: 13 May 2012

Abstract Despite the long history of the cell formation problem (CF) and availability of dozens of approaches, very few of them explicitly optimize the objective of cell formation. These scarce approaches usually lead to intractable formulations that can be solved only heuristically for practical instances. In contrast, we show that CF can be explicitly modelled via the minimum multicut problem and solved to optimality in practice (for moderately sized instances). We consider several real-world constraints that can be included into the proposed formulations and provide experimental results with real manufacturing data.

Keywords Cell formation · Group technology · Minimum multicut

Mathematics Subject Classification 05C70 · 90C35 · 90C11

1 Introduction

Cell formation (CF) is a key step in implementation of group technology—a paradigm in industrial engineering developed by Mitrofanov (1966) and Burbidge (1961), and

D. Krushinsky (✉)
Department of Operations, University of Groningen,
P. O. Box 800, 9700 AV Groningen, The Netherlands
e-mail: d.krushinsky@rug.nl

B. Goldengorin
LATNA, Laboratory of Algorithms and Technologies for Networks Analysis
and Department of Applied Mathematics and Informatics, Nizhny Novgorod branch
of The National Research University Higher School of Economics,
B. Pecherskaya, 25/12, 603155 Nizhny Novgorod, Russia
e-mail: bgoldengorin@hse.ru

suggesting that similar parts should be processed in a similar way. In the most general setting, the (unconstrained) CF problem can be formulated as follows. Given finite sets of machines and parts that must be processed within a certain time period, the objective is to group machines into manufacturing cells (hence the name of the problem) and parts into the product families such that each product family is processed mainly within one cell. Equivalently, this objective can be reformulated as minimization of what is usually referred to as the amount of *intercell movement*—the flow of parts travelling between the cells. This amount can be expressed via the number of parts, their total volume or mass, depending on the particular motivation for CF. For example, if cells are spatially distributed it may become important to reduce transportation costs that depend on the mass or volume rather than on the number of parts.

Throughout the decades the problem has gained a lot of attention resulting in hundreds of papers and dozens of approaches that use all the variety of tools ranging from intuitive iterative methods (e.g., McCormick et al. 1972; King 1980; Wei and Kern 1989) to neural networks (e.g., Kaparthi and Suresh 1992; Yang and Yang 2008), evolutionary algorithms (e.g., Adil and Rajamani 2000; Filho and Tiberti 2006) and mixed-integer programming (e.g., Chen and Heragu 1999; Bhatnagar and Saddikuti 2010); an overview can be found in Selim et al. (1998). Despite all this variety, to the best of our knowledge, there is no tractable approach that explicitly optimizes the mentioned above goal. In particular, all the available approaches have at least one of the following drawbacks:

- the model itself is an approximation to the original problem;
- the model is solved by a heuristic procedure.

To illustrate the first point we would like to mention that it is a common practice to reduce the size of the problem by considering only relations between machines instead of considering machine-part relations. Such a framework is quite beneficial due to the fact that the number of machines is quite limited (usually less than 100) while the number of parts can be magnitudes larger. This point will be clearly illustrated below by means of an industrial example. The reduction is usually implemented by introducing a machine–machine similarity measure that can be based on the similarity of sets of parts that are processed by a pair of machines, on similarity of manufacturing sequences of these parts, etc. Literature reports several similarity measures, an overview can be found in Yin and Yasuda (2006). However, all of them are based on intuitive considerations and there is no strict reasoning why one of them is better than another. If such an inexact similarity measure is further plugged into some model, then the whole model is nothing more than an approximation to the original problem. Finally, the resulting model often appears to be NP-hard and its authors are forced to use heuristic solution methods further deteriorating the solution quality.

The purpose of this paper is to formulate an exact model for the CF problem, flexible enough to allow additional practically motivated constraints and solvable in acceptable time at least for moderately sized realistic instances.

The paper is organized as follows. In the next section we discuss the exact model for cell formation, show that it is equivalent to the minimum multicut problem and discuss its computational complexity. In Sects. 3 and 4 we motivate and present two MILP formulations for the problem. Section 5 is focused on additional constraints that

may be introduced into the model, while Sect. 6 provides results of experiments with real manufacturing data. Section 7 summarizes the paper with a brief discussion of the obtained results and further research directions.

2 The essence of the cell formation problem

In this section we formalize the CF problem given two types of the input data and show how it can be modelled via the minimum multicut problem. For the rest of this paper let sets $I = \{1, \ldots, m\}$ and $J = \{1, \ldots, r\}$ enumerate machines and parts, respectively, and let p denote the number of cells.

Quite often, the input data for the CF problem is given by an $m \times r$ binary machine-part incidence matrix (MPIM) $\mathbf{A} = [a_{ij}]$ where $a_{ij} = 1$ only if part j needs among others machine i, see Fig. 1a. Given such an input, the problem is equivalent (see, e.g., Burbidge 1991) to finding independent permutations of rows and columns that turn $\mathbf{A}$ to an (almost) block-diagonal form and minimize the number of out-of-block ones, also known as *exceptional elements*. The diagonal blocks correspond to cells, and the number of exceptional elements reflects the amount of intercell movement, see Fig. 1b.

Given such an interpretation, the problem is similar to the biclustering problem (see, e.g., Madeira and Oliveira 2004). Though for the general biclustering problem there exist efficient exact methods (see, e.g., DiMaggio et al. 2008), they are hardly applicable to CF because most of them allow each row or column to belong to more than one cluster (see, e.g., Madeira and Oliveira 2004), while for CF the issue of non-overlapping blocks is critical. In addition, as we show further in this section, block-diagonalisation does not exactly minimise the intercell movement as it ignores operational sequences.

Though the well known block-diagonal interpretation is easy to perceive, we will consider the problem from a completely different, yet insightful, viewpoint. Without any loss of generality one can associate with matrix $\mathbf{A}$ an undirected bipartite graph $G(I \cup J, E)$ by simply treating $\mathbf{A}$ as an incidence matrix of G. Note, that such an interpretation was also considered for the biclustering problem. Taking into account that each nonzero element of $\mathbf{A}$ corresponds to an edge in G, it is not hard

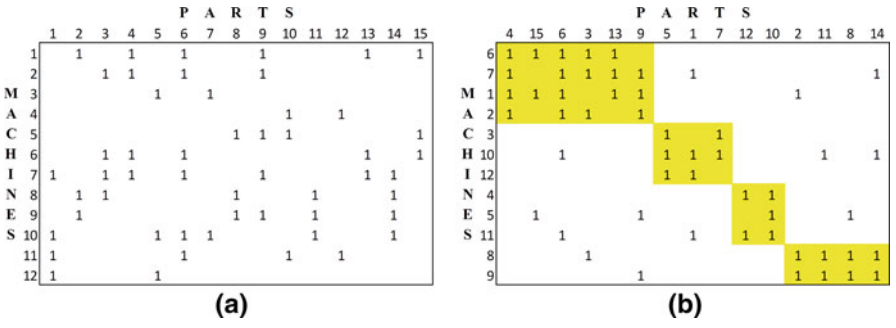

Fig. 1 An example of a machine-part incidence matrix (MPIM): **a** raw data, **b** block-diagonalized form (blocks are highlighted). Zero entries are not shown for clarity

to understand that diagonal blocks of $\mathbf{A}$ correspond to disjoint nonempty subgraphs $G_1, \ldots, G_p$ of G. Consider now the set of edges E' corresponding to exceptional elements and observe that each edge from E' has its endpoints in different subgraphs G_i , $i \in \{1, \ldots, p\}$. Thus, E' can be thought of as a cut that splits G into p nonempty subgraphs. Further, we call a cut with this property a *p-cut*. Assuming that all edges of G have a unit weight and taking into account the relation between E' and exceptional elements, it is possible to reformulate the CF problem in terms of graphs as follows: given an undirected weighted graph find a p-cut of the minimum weight. Let us abbreviate this problem as MINpCUT, in literature it is also known as "min k-cut" (we prefer to denote the number of subgraphs by p as letter k is handy as an index).

One may notice that the MINpCUT based approach has a negative feature as compared to many other models. Instead of using machine–machine relations it works directly with machine-part data, i.e. a MINpCUT instance can be very large (G may have thousands of vertices) and there is no straightforward way to overcome this. However, we argue that this shortcoming is induced by the "inadequate" format of input data rather than by the model itself. Indeed, irrespectively of the solution approach, the MPIM does not contain enough information to correctly handle the following aspects:

- distinguish between the following two cases:
  (a) a part is processed in one cell and then in the second cell;
  (b) a part is processed in one cell, then in the second cell and then again in the first one;
- a part visits some machines several times, i.e. its manufacturing sequence looks like . . .–M1–M2–M1–M2–. . . (this may correspond, for example, to cycles of thermal processing).

Thus, all approaches using the machine-part incidence matrix as an input (e.g., the one from Chen and Heragu 1999) solve only approximation of the original problem, quite often by a heuristic. In addition, the common practice of deriving machine-machine relations from a MPIM looks somewhat awkward from the methodological point of view. It seems more logical to derive these relations directly from the manufacturing data normally containing more information, for example, the sequence in which machines are visited by each part.

The above mentioned considerations motivated us to reconsider the essence of the cell formation problem. As mentioned above, the objective is to minimize the parts flow between cells. The latter quantity is nothing else than the parts flow between two machines summed up for all pairs of machines belonging to different cells. In terms of graphs this can be expressed as follows. Consider a weighted graph $G(I, E)$, where each vertex corresponds to a machine. An edge $(i, j) \in E$ is assigned a weight equal to the amount of parts going directly from machine i to j and in the opposite direction. Clearly, a p-cut in such a graph produces p machine cells and its weight is equal to the amount of intercell movement that must be minimized. Once the machine cells are generated, part families can be compiled by assigning each part to a machine cell performing most operations on it. Thus, we again end up with the MINpCUT problem, but now it is defined on a graph that has only m vertices,

as compared to $m + r$ vertices in case of input data given by a machine-part incidence matrix. We would like to mention that somewhat similar considerations about the graph-theoretic origins of the exact model for CF can be found in Boulif and Atif (2006), however, authors do not mention its relation to the min multicut problem, nor provide evidence of tractability for their approach. Another graph-related approaches to CF include those based on the minimum spanning tree (MST; see, e.g., Ng 1993) and the p-median (PMP; see, e.g., Won and Lee 2004) problems. Their difference from our approach can be made clearer by observing that by minimising a p-cut one maximises the total weight of edges within p subgraphs. Instead of optimising all weights within subgraphs, MST and PMP-based approaches consider only those falling within a certain pattern: a spanning tree or a tree of depth 1, respectively.

Once we know that the cell formation problem is equivalent to MINpCUT, we may analyse its complexity based on the properties of the latter. First of all, consider the case $p = 2$. MIN2CUT is a straightforward generalization of the well-known min $s - t$ cut problem where optimization is to be done for all pairs (s, t). A closer view makes it possible to conclude that for a graph $G(V, E)$ it is enough to solve $|V| - 1$ min $s - t$ cut instances. As the minimum 2-cut (as well as any 2-cut) splits G into 2 subgraphs, one can fix s lying in one of them and iterate through all possible vertices t until the one lying in the other subgraph is found. Thus, in case of two cells the CF problem without additional constraints is polynomially solvable. On the other hand, as p gets close to $|V|$ the problem becomes easy as well. For example, if $p = |V| - 1$ there is exactly one pair of vertices that must be placed in one subgraph (other $p - 1$ subgraphs are just singletons). Further, if $p = |V| - 2$ there are either two pairs or one triple of vertices that must not be disconnected by a cut. This intuition can be extended further and it becomes clear that the combinatorial complexity of the problem quickly increases as p tends to $|V|/2$.

In a general case the problem is NP-hard, having a polynomial complexity $O(n^{p^2} T(n))$ for fixed p (Goldschmidt and Hochbaum 1994); $T(n)$ denotes time for solving one min $s - t$ cut problem for a graph with n vertices. For a particular case $p = 3$ there also exists an efficient $O(mn^3)$ algorithm by Burlet and Goldschmidt (1997), where n and m are numbers of vertices and edges, respectively. A number of approximate algorithms are known (see, e.g., Saran and Vazirani 1995; Ravi and Sinha 2008) with the best approximation ratio being $(2 - 2/p)$ (Saran and Vazirani 1995).

Thus, for $p = 2, 3, |V| - 2, |V| - 1$ the MINpCUT problem (therefore, the unconstrained CF problem) can be efficiently solved even for large instances, while becoming computationally intractable as p gets closer to $|V|/2$. Most papers on MINpCUT propose specialized algorithms, not allowing additional constraints to be involved and thus inapplicable to CF. This lack of flexible approaches motivated us to develop MILP formulations that can be extended by any linear constraints and solved using a general-purpose solver (at least, for moderately sized instances).

3 MINpCUT: a straightforward formulation (SF)

In this section we present and discuss a straightforward formulation (SF) of MINpCUT problem that will be further used in the numerical experiments. Let $G(V, E)$ be an undirected weighted graph with $|V| = n$ vertices, let c_{ij} denote the weight

of edge $(i, j) \in E$. For the rest of the paper, let indices i and j enumerate vertices, $i, j \in \{1, \ldots, n\}$, and index k enumerate subgraphs, $k \in \{1, \ldots, p\}$; constant S is defined as a sum of all edge weights:

$$S = \sum_i \sum_{j>i} c_{ij}. \tag{1}$$

SF uses two sets of variables: v_{ik} variables reflecting assignment of vertices to subgraphs and z_{ijk} variables reflecting assignment of pairs of vertices i and j to subgraphs k. Under the introduced notations the SF formulation can be written as follows:

$$S - \sum_i \sum_{j>i} \sum_k c_{ij} z_{ijk} \longrightarrow \min \tag{2}$$

$$\sum_i v_{ik} \geq 1 \quad \forall k \tag{3}$$

$$\sum_k v_{ik} = 1 \quad \forall i \tag{4}$$

$$z_{ijk} \leq v_{ik} \quad \forall i \neq j, k \tag{5}$$

$$z_{ijk} \leq v_{jk} \quad \forall i \neq j, k \tag{6}$$

$$z_{ijk} \geq v_{ik} + v_{jk} - 1 \quad \forall i \neq j, k \tag{7}$$

$$v_{ik} \in \{0, 1\} \quad \forall i, k \tag{8}$$

$$z_{ijk} \in [0, \infty) \quad \forall i \neq j, k\ . \tag{9}$$

The objective (2) minimizes the difference between the sum of all edge weights and the sum of weights of the edges within subgraphs, i.e. the weight of the p-cut. Constraints (3) ensure that each subgraph has at least one vertex, i.e. there are exactly p nonempty subgraphs. Constraints (4) ensure that each vertex is included into exactly one subgraph. Finally, constraints (5)–(7) are needed to guarantee that a pair of vertices i and j are assigned to the subgraph k if and only if each of them is assigned to subgraph k. The formulation uses $n \times p$ Boolean v-variables, while for z-variables nonnegativity is sufficient as constraints (5)–(7) force them to take Boolean values.

It is easy to see that the formulation SF has the following property: the number of variables and, therefore, complexity increases with increasing p. Though for small p the formulation is rather efficient (as will be shown in Sect. 6) for larger values of p it becomes intractable. It should be noted that SF does not reflect the fundamental property of the problem: tractability for both small and large (close to n) values of p. This observation motivated us to develop an alternative formulation that will reflect the problem complexity more adequately.

4 MINpCUT: an alternative formulation (AF)

Without any loss of generality one can think of G as of a complete graph with some edges (those not actually present) having zero weight. Under this assumption of

completeness, a p-cut decomposes G into p subcliques, leading to the following properties of the feasible solutions. First of all, for any three vertices presence of any two edges between them induces presence of the whole triangle on these vertices. If one calls two edges having a vertex in common *adjacent edges*, then the property can be expressed as follows: each pair of adjacent edges induces the third edge adjacent to both of them. The next property is that any particular vertex in a subclique is connected to any other vertex in a subclique. These two simple properties play an important role in our formulation AF. It uses the following Boolean variables: x_{ij} is nonzero only if edge (i, j) is not removed by a p-cut, and y_i is nonzero only if vertex i is selected as a special vertex. Each vertex in a subclique can be selected as a special vertex, and exactly one vertex in a subclique is special. This setting is needed to count subcliques. The rest of notations are preserved from the previous sections, and AF can be expressed as:

$$S - \sum_i \sum_{j>i} c_{ij} x_{ij} \longrightarrow \min \tag{10}$$

$$\sum_i y_i = p \tag{11}$$

$$x_{ij} \leq 2 - y_i - y_j \quad \forall i \neq j \tag{12}$$

$$x_{ij} \geq x_{il} + x_{jl} - 1 \quad \forall i \neq j \neq l \tag{13}$$

$$x_{ij} = x_{ji} \quad \forall i \neq j \tag{14}$$

$$y_i \in \{0, 1\} \quad \forall i \tag{15}$$

$$x_{ij} \in \{0, 1\} \quad \forall i \neq j\ . \tag{16}$$

Similarly to SF, the objective (10) minimizes the difference between the sum of all edge weights and the sum of weights of the edges within subcliques, i.e. the weight of the p-cut. Constraint (11) ensures that exactly p special vertices must be selected, while constraints (12) force each pair of special vertices to be disconnected, such that each subclique contains a single special vertex. Constraints (13) ensure the mentioned above property: any two adjacent edges force the third adjacent edge to be preserved. Finally, constraints (14) preserve undirected structure of the problem. It is not hard to understand that these constraints can be used to eliminate half of the x-variables, i.e. to use only those x_{ij} for which $i < j$ holds. In our experiments we used such a reduced formulation.

5 Additional constraints

Though the MINpCUT based model exactly optimizes the objective of cell formation, additional constraints ensuring practical feasibility of obtained solutions are usually needed. Moreover, it is desirable to be able to take into account additional factors and managerial preferences. As our model has quite a general structure, in principle, any constraints that can be expressed in a linear form can be included. In this section we give some examples of extending our formulations SF and AF with additional constraints.

First of all, some flexibility in the model is provided by weights c_{ij}. It is not hard to understand that these values can be defined either as the number of parts travelling between machines i and j, or their total mass, volume, etc. However, the range of possible factors is not limited to properties of parts. For example, it may be desirable to account for the available workforce and reduce the so-called cross-training costs (see, e.g., Bhatnagar and Saddikuti 2010). In this case, the objective is to ensure that each worker is able to deal with as much machines in his cell as possible. This issue can be modelled by making weights c_{ij} dependent on the number of workers able to operate both machines i and j.

The next issue that can be easily dealt with is based on the fact that some machines cannot be placed in the same cell (e.g., because of safety reasons) while others must be placed close to each other because of managerial considerations or constructional peculiarities. In both formulations SF and AF it is easy to force a pair of machines i and j to be grouped in one cell or in different cells. In case of SF, the constraints

$$v_{ik} = v_{jk} \quad \forall k \tag{17}$$

and

$$v_{ik} + v_{jk} \leq 1 \quad \forall k \tag{18}$$

force or prohibit assignment of machines i and j to the same cell, respectively. For AF the corresponding constraints are

$$x_{ij} = 1 \tag{19}$$

and

$$x_{ij} = 0, \tag{20}$$

leading to a problem with fewer Boolean variables (as some x-variables become fixed).

Capacity constraints are, probably, the most popular ones in cell formation; these set a limit on the minimum or maximum number of machines in a cell. Indeed, there is little sense in cells containing a single machine, while such solutions are common for the manufacturing data that we experienced. In order to limit the number of machines per cell from below by n_L, SF must be equipped with the following constraints:

$$\sum_i v_{ik} \geq n_L \quad \forall k \ . \tag{21}$$

For AF the constraints look like

$$\sum_{j,\, j \neq i} x_{ij} + 1 \geq n_L \quad \forall i \ . \tag{22}$$

Validity of these constraints can be expressed by the fact that each vertex is connected to all other vertices within its subclique. Thus, the number of incident edges

not removed by a p-cut plus the vertex i itself is equal to the total number of vertices in a subclique. It should be mentioned that the system of constraints (22) is redundant in a sense that p constraints written for vertices i all lying in different subcliques are sufficient. However, it is not known beforehand which p vertices will belong to different subcliques in an optimal solution (these are determined by y-variables). Upper bounds on the number of machines per cell can be set in a similar way.

Workload balancing constraints are used to ensure that cells have a balanced load in terms of working hours, so that the tasks are evenly divided between the cells. Such balancing helps to avoid the situation when one cell (and the corresponding team of workers) is overloaded, while another is underutilised. If one denotes by w_i the workload of machine i and by w_L the lower bound on the workload per cell, then constraints for SF and AF become

$$\sum_i w_i v_{ik} \geq w_L \quad \forall k \tag{23}$$

and

$$\sum_{j,\, j \neq i} w_j x_{ij} + w_i \geq w_L \quad \forall i \ , \tag{24}$$

respectively. It can be seen that the workload balancing and capacity constraints have very similar structure.

In the rest of this section we discuss a much less trivial issue—the presence of identical machines. It is not uncommon, especially in large manufacturing systems, that some most extensively used machines are present in several copies. This means that each part can be processed on either of these machines equally well and if one applies any clustering algorithm directly, the identical machines will always be grouped together. On the other hand, placing them in different cells reduces intercell movement (if they are needed in more than one cell). This issue is usually hard to model as it leads to the so-called disjunctive constraints—a part can be processed on either of the identical machines. However, here we show that our formulations can be adjusted to account for identical machines without significant complication.

Note that in the above discussion we could have used the term "machine types" instead of "machines", implicitly assuming that identical machines are placed together. Let us call placement of identical machines in different cells *separation of identical machines*. Now we are going to modify the formulations SF and AF such that they allow separation of identical machines; these will be denoted as SFs and AFs, respectively.

Let us denote by n_i the number of identical machines of type i. Recall that indices i and j enumerate machine types , $i, j \in \{1, \ldots, n\}$, and index k enumerates cells or subgraphs, $k \in \{1, \ldots, p\}$. The modification SFs of formulation SF can be written as follows:

$$S - \sum_i \sum_{j>i} c_{ij} x_{ij} \longrightarrow \min \tag{25}$$

$$\sum_i v_{ik} \geq 1 \quad \forall k \tag{26}$$

$$\sum_k v_{ik} \geq 1 \quad \forall i \tag{27}$$

$$\sum_k v_{ik} \leq n_i \quad \forall i \tag{28}$$

$$z_{ijk} \leq v_{ik} \quad \forall i \neq j, k \tag{29}$$

$$z_{ijk} \leq v_{jk} \quad \forall i \neq j, k \tag{30}$$

$$z_{ijk} \geq v_{ik} + v_{jk} - 1 \quad \forall i \neq j, k \tag{31}$$

$$x_{ij} \leq \sum_k z_{ijk} \quad \forall i \neq j \tag{32}$$

$$x_{ij} \leq 1 \quad \forall i \neq j \tag{33}$$

$$v_{ij} \in \{0, 1\} \quad \forall i \neq j \tag{34}$$

$$z_{ijk} \in [0, \infty) \quad \forall i \neq j, k \tag{35}$$

$$x_{ij} \in [0, \infty) \quad \forall i \neq j\ , \tag{36}$$

where v-, x- and z-variables have the same meaning as in Sects. 3 and 4. The objective (25) minimizes the total weight of the edges removed by a p-cut, and constraints (26), (29)–(31) and (34)–(35) are inherited from SF. Constraints (27)–(28) are a generalization of constraint (4). These require each vertex (machine type) i to be included into at least one subgraph (cell) and at most n_i subgraphs (at most n_i machines of type i are used). Finally, constraints (32)–(33) cut the edge between i and j if this pair of vertices is not contained in any of p subgraphs and ensure that each edge can be cut only once. It can be seen that these constraints also force x-variables to take 0–1 values and the numbers of Boolean variables in formulations SF and SFs are equal.

The modification of the formulation AF to allow separation of the machines is even simpler than in case of SF. This task can be accomplished by considering a graph where each vertex corresponds to a single machine (not to a machine type, as in case of SFs) and penalising the objective such that vertices corresponding to identical machines are forced to be assigned to different subcliques. The penalising term for any pair of identical machines i and i' can be written as

$$+\left[\sum_j c_{ij}\right] x_{ii'}, \tag{37}$$

where the constant in brackets is large enough to ensure that the negative impact of placing vertices i and i' into one subclique cannot be compensated by any arrangement of other vertices. Instead of penalising the objective, one may also add the following constraint:

$$x_{ii'} = 0\ . \tag{38}$$

However, such constraints may conflict with capacity or other constraints leading to an infeasible problem. Thus, AFs inherits the structure of AF and has few additional constraints or terms in the objective function.

In conclusion, we would like to mention that if for some machine type i holds $n_i \geq p$, then it can be excluded from consideration as a machine of this type can be added to each cell. In particular, this implies that in case of two cells identical machines make the problem smaller, therefore easier.

6 Computational experiments

In order to motivate the exact model for cell formation we considered several used in literature instances that were tackled by heuristic approaches. The scope of this study was restricted to the instances containing the operations sequence data and to the papers reporting complete solutions (assignment of machines to cells) so that the amount of intercell movement can be estimated. In order to ensure the most consistent comparison, we restricted the number of machines per cell both from below and from above by the values inherent to the corresponding solutions from the literature. The computational results are summarised in Table 1, where the first two columns indicate the number of machines, parts, and the number of cells to be made. The next column indicates the amount of intercell movement achieved by our MINpCUT based model. The following two columns contain the best result we could find in the literature and a corresponding reference; the last column reports running times for SF. As can be seen from Table 1, in most cases contemporary heuristics were unable to find optimal solutions. At the same time, running times for our model are quite limited, except the last considered instance which we could not solve to optimality. In these and the following experiments we used a moderate PC (Intel Core2 Duo, 2.33 GHz, 2 GB

Table 1 Performance comparison with heuristic approaches from literature in terms of intercell movement

Size	p	Our result	Best known	Source	Time (s)
8×20	3	17	17	Nair and Narendran (1998)	<1
12×19	2	9	16	Ahi et al. (2009)	<1
12×19	3	20	27	Ahi et al. (2009)	<1
18×35	4	47	54	Ahi et al. (2009)	7
20×20	5	17	18	Ahi et al. (2009)	10
20×51	5	36	36	Ahi et al. (2009)	17
20×20	5	18	19	Nair and Narendran (1998)	13
25×40	4	17	22	Ahi et al. (2009)	8
25×40	6	27	45	Ahi et al. (2009)	140
25×40	8	56	72	Nair and Narendran (1998)	$\sim$ 24 h[a]

[a] Interrupted due to memory limitations, best integer solution is reported (best lower bound is 50.963). In fact, the reported solution was found within 1 h, the rest of the time was spent on tightening the lower bound

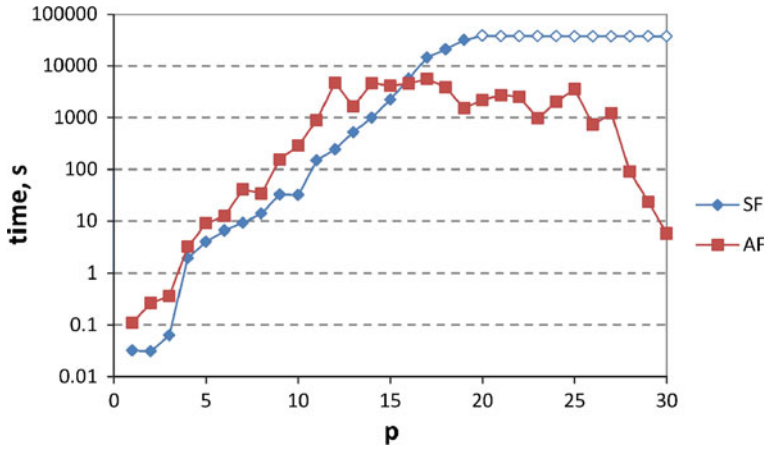

Fig. 2 Solution times for an instance with 30 machines (limited by 10 h)

RAM) and Xpress-MP as a MILP solver. The solver was restricted to use one processor core.

The aim of our further experiments was to check computational properties of the introduced model and to show its practical applicability by means of an industrial case. As a testbed for the experiments we considered data from a small company producing high precision tools. The quantitative characteristics of the dataset are as follows:

- time period: 11 months;
- 30 machine types;
- 7563 part types;
- 25080 operations (4149 part moves between machines).

First, we tried to solve the unconstrained CF problem for all possible values of p using both our formulations. The results are summarised in Fig. 2 (the running time was limited by 10 h). As predicted in Sect. 3, the running times for SF grow with increasing p, while AF is efficient for p close to 1 and to n. Note that the instance with 30 machines for p up to 15 can be solved within an hour, which is reasonable taking into account that cells are not reconfigured every day. Note also that the size of the instance under consideration is quite substantial: after reviewing dozens of papers on cell formation we were able to find only 3 realistic instances with more than 30 machines (the largest one having 50 machines). At the same time, the number of parts does not affect the performance of our model.

Next, we conducted several experiments in order to demonstrate the issue of identical machines and its possible impact. Figure 3 shows the intercell movement, expressed as a percentage of the total parts movement, for $p = 2$ cells and different lower bounds on the number of machines per cell. It can be seen that if possibility of separating identical machines is ignored, balanced cells are impossible due to a large intercell movement of 18.29 %. In the opposite case, two reasonable cells can be obtained with only 0.19 % intercell movement. In case of three cells, the corresponding figures are 24.13 and 1.16 %. Figures 4 and 5 illustrate the obtained cellular decompositions, matrices display the numbers of parts travelling directly between each pair of machines.

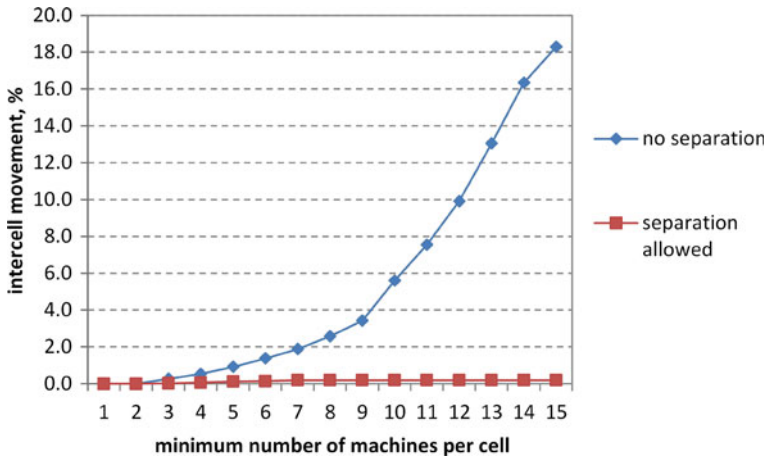

Fig. 3 Intercell movement for different restrictions on the number of machines per cell for the case of two cells

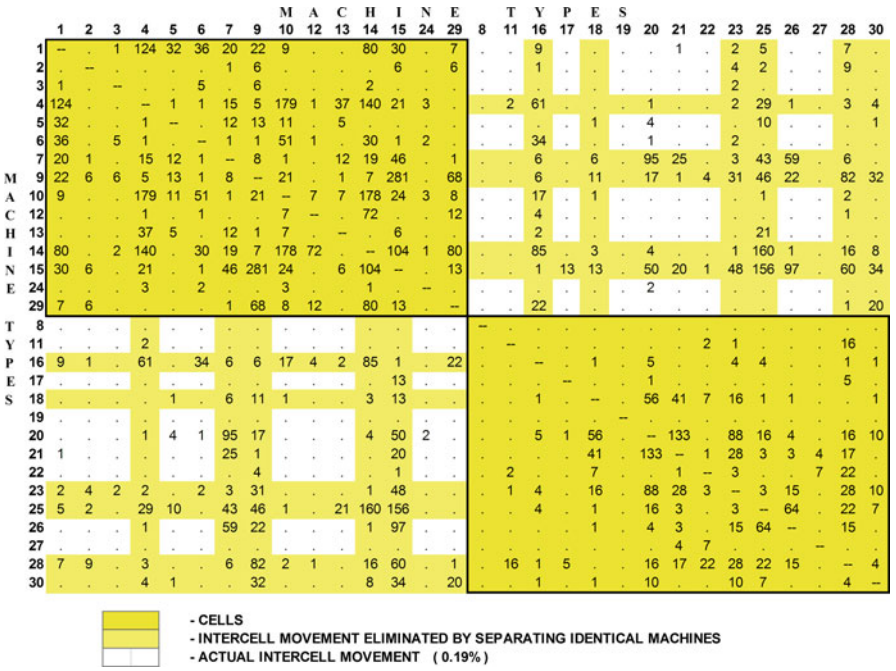

Fig. 4 An optimal decomposition into two cells (zero entries are denoted by *dots*)

7 Summary

In this paper an exact model for the cell formation problem in group technology is developed. We have demonstrated that a machine-parts incidence matrix does not contain enough information to obtain truly optimal solutions. As becomes apparent from

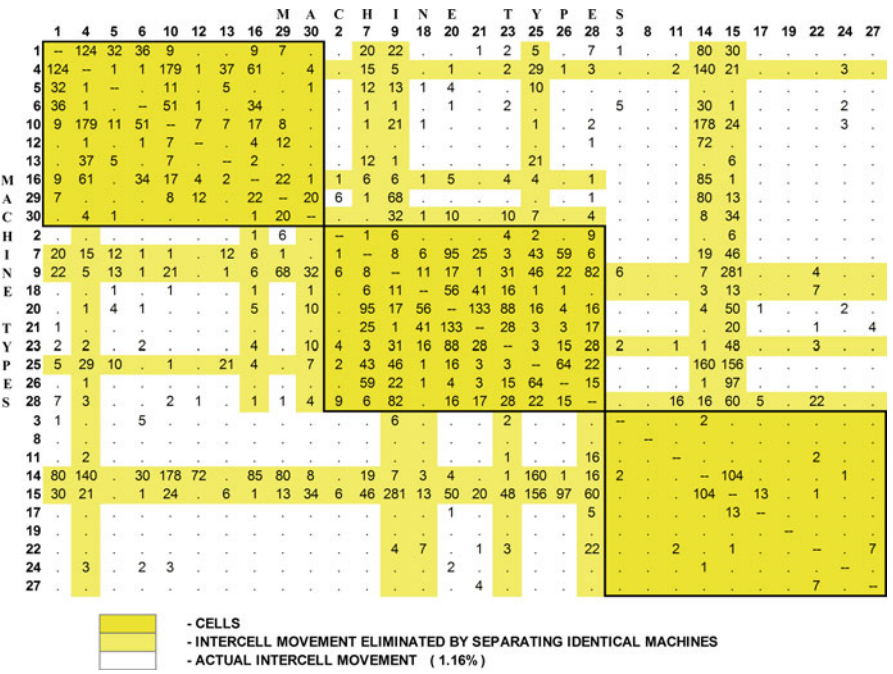

Fig. 5 An optimal decomposition into three cells (zero entries are denoted by *dots*)

the presented experimental comparison, recent heuristics taking operational sequences into account usually lead to suboptimal solutions. We have also shown that an exact model can be independent of the number of parts and demonstrated importance of this property by an industrial example.

It was shown that the cell formation problem with a fixed number of cells is equivalent to the minimum multicut problem (also, if the input data is given by a machine-part incidence matrix). This fact immediately implies polynomial solvability of the former in case $p = 2$. For an arbitrary number of cells, however, the problem remains NP-hard. Yet, it can still be solved to optimality in many practical cases due to the limited number of machines in real manufacturing systems. If the instance is too large to be solved optimally, the following iterative heuristic procedure can be used. For the initial problem solve the minimum 2-cut problem, then iteratively pick the largest cell and solve for it the minimum 2-cut until p cells are obtained. It is easy to show that whenever (almost) independent cells are possible the obtained solution will be globally optimal. Optimality conditions for this or another heuristics may become possible directions for future research.

We presented two MILP formulations that we call SF and AF, and demonstrated their tractability for moderately sized instances by means of an industrial example. It was found that SF is more efficient than AF for small values of p, becoming intractable for larger ones. At the same time, AF performs well for values of p close to 1 and to the number of vertices in a graph. The latter formulation better reflects structure

of the problem and, potentially, may be more suitable for size reduction based on graph-theoretic considerations.

We considered several additional constraints ranging from very popular capacity constraints to the particular case of disjunctive constraints induced by identical machines. To the best of our knowledge, there are no models adequately handling identical machines, even though some attempts are reported in literature. In contrast, we have shown that identical machines can be modelled in both our formulations and demonstrated how it works by means of an industrial example. Overall, our numerical experiments confirmed practical applicability of the proposed model for real-life problems with a moderate number of machine types.

Possible directions for the further research may include development of advanced formulations for the minimum multicut problem, and/or problem size reduction approaches. Such directions may be interesting taking into account that the problem has quite a general clustering nature and its possible applications are not limited to cell formation in group technology.

Acknowledgments The authors would like to thank prof. Jannes Slomp (University of Groningen, Department of Operations) for his help in obtaining industrial data and suggestion to consider the issue of identical machines. Boris Goldengorin is partially supported by LATNA Laboratory, NRU HSE, RF government grant, ag. 11.G34.31.0057 and the grant "Teachers-Students" No. 11-04-0008 "Calculus for tolerances in combinatorial optimization: theory and algorithms".

References

Adil GK, Rajamani D (2000) The tradeoff between intracell and intercell moves in group technology cell formation. J Manuf Syst 19(5):305–317

Ahi A, Aryanezhad M, Ashtiani B, Makui A (2009) A novel approach to determine cell formation, intracellular machine layout and cell layout in the CMS problem based on TOPSIS method. Comput Oper Res 36(5):1478–1496. doi:10.1016/j.cor.2008.02.012

Bhatnagar R, Saddikuti V (2010) Models for cellular manufacturing systems design: matching processing requirements and operator capabilities. J Opl Res Soc 61:827–839. doi:10.1057/jors.2008.181

Boulif M, Atif K (2006) An exact multiobjective epsilon-constraint approach for the manufacturing cell formation problem. In: Proceedings of the international conference on service systems and service management, vol 2, pp 883–888. doi:10.1109/ICSSSM.2006.320737

Burbidge JL (1961) The new approach to production. Prod Eng 40:3–19

Burbidge JL (1991) Production flow analysis for planning group technology. J Oper Manag 10(1): 5–27. doi:10.1016/0272-6963(91)90033-T

Burlet M, Goldschmidt O (1997) A new and improved algorithm for the 3-cut problem. Oper Res Lett 21:225–227

Chen JS, Heragu SS (1999) Stepwise decomposition approaches for large scale cell formation problems. Eur J Oper Res 113:64–79

DiMaggio PA, McAllister SR, Floudas CA, Feng XJ, Rabinowitz JD, Rabitz HA (2008) Biclustering via optimal re-ordering of data matrices in systems biology: rigorous methods and comparative studies. BMC Bioinformatics 9(1):458

Filho EVG, Tiberti AJ (2006) A group genetic algorithm for the machine cell formation problem. Int J Prod Econ 102:1–21

Goldschmidt O, Hochbaum DS (1994) A polynomial algorithm for the k-cut problem for fixed k. Math Oper Res 19(1):24–37

Kaparthi S, Suresh NC (1992) Machine-component cell formation in group technology: a neural network approach. Int J Prod Res 30(6):1353–1367
King JR (1980) Machine-component grouping in production flow analysis: an approach using a rank order clusterng algorithm. Int J Prod Res 18(2):213–232
Madeira SC, Oliveira AL (2004) Biclustering algorithms for biological data analysis: a survey. IEEE-ACM T Comput BI 1(1):24–45
McCormick WT, Schweitzer PJ, White TW (1972) Problem decomposition and data reorganization by a clustering technique. Oper Res 20(5):993–1009. doi:10.1287/opre.20.5.993
Mitrofanov SP (1966) Scientific principles of group technology, Part I. National Lending Library of Science and Technology, Boston
Nair GJ, Narendran TT (1998) CASE: a clustering algorithm for cell formation with sequence data. Int J Prod Res 36(1):157–180
Ng SM (1993) Worst-case analysis of an algorithm for cellular manufacturing. Eur J Oper Res 69:384–398
Ravi R, Sinha A (2008) Approximating k-cuts using network strength as a lagrangean relaxation. Eur J Oper Res 186:77–90
Saran H, Vazirani V (1995) Finding k-cuts within twice the optimal. SIAM J Comput 24(1):101–108
Selim HM, Askin RG, Vakharia AJ (1998) Cell formation in group technology: review, evaluation and directions for future research. Comput Ind Eng 34(1):3–20
Wei JC, Kern GM (1989) Commonality analysis. A linear cell clustering algorithm for group technology. Int J Prod Res 27(12):2053–2062
Won Y, Lee KC (2004) Modified p-median approach for efficient GT cell formation. Comput Ind Eng 46(3):495–510. doi:10.1016/j.cie.2004.01.010
Yang MS, Yang JH (2008) Machine-part cell formation in group technology using a modified ART1 method. Eur J Oper Res 188(1):140–152. doi:10.1016/j.ejor.2007.03.047
Yin Y, Yasuda K (2006) Similarity coefficient methods applied to the cell formation problem: a taxonomy and review. Int J Prod Econ 101(2):329–352. doi:10.1016/j.ijpe.2005.01.014