SCHEDAE INFORMATICAE

VOLUME 20 2011

On the number of clusterings in a hierarchical classification model with overlapping clusters

Adam Roman, Igor T. Podolak, Agnieszka Deszyńska
Institute of Computer Science, Jagiellonian University,
Prof. Stanisława Łojasiewicza 6, 30–348 Cracow, Poland
e-mail: {*roman,podolak*}*@ii.uj.edu.pl*, *adeszynska@gmail.com*

Abstract. This paper shows a new combinatorial problem which emerged from studies on an artificial intelligence classification model of a hierarchical classifier. We introduce the notion of proper clustering and show how to count their number in a special case when 3 clusters are allowed. An algorithm that generates all clusterings is given. We also show that the proposed approach can be generalized to any number of clusters, and can be automatized. Finally, we show the relationship between the problem of counting clusterings and the Dedekind problem.

1. Motivation

In machine learning, a classifier Cl has to assign an input attribute vector att to one class from a predefined set $\mathcal{K} = \{1, 2, \ldots, K\}$. Such a classifier is built using a training set consisting of examples from set of pairs $D = \{(att_i, x_i)\}_{i=1}^{N}$, where x_i is the correct classification. Several approaches exist which use neural networks, decision trees, etc. One possibility is to combine results from several simple classifiers Cl_i, which may be *weak*, *i.e.* classify only slightly better than a random classifier. For a simple two–class problem, this would require correct classification of $1/2+\epsilon$ fraction of examples, while for a K–class problem, this would be, roughly speaking, above $1/K$ fraction of examples (it depends on the actual measure used). Such combination may provide a classifier that would classify correctly almost all examples. One well-known algorithm is the Adaboost which builds subsequent weak classifiers by training them on training sets built from the original with example distribution changed [1, 2].

We have proposed a different approach to the above problem by means of a Hierarchical Classifier (HC) algorithm in which the training set is, upon building subsequent classifiers, divided into overlapping subsets, which define subproblems to be solved [3].

DEFINITION 1. *For a training set* $D = \{(att_i, x_i)\}_{i=1}^{N}$, $x_i \in \mathcal{K} = \{1, 2, \ldots, K\}$, *where* $\mathcal{K}$ *is the set of classes, a Hierarchical Classifier HC is defined as a tree structure*

1. *the root classifier* Cl_0 *is a weak classifier into* K *classes*
2. *a clustering algorithm groups classes which were similarly classified by* Cl_0 *into set of* J *clusters* $\mathcal{C} = \{C_1, \ldots, C_J\}$, $C_j \subset \mathcal{K}$,
3. *for each cluster* C_j
   - *a new training set* $D_j = \{(att_i, x_i) \in D : \ x_i \in C_j\}$ *is extracted from the original set* D
   - *a new classifier* Cl_j *is built in the same way.*

HC may be built recursively until a low error is achieved. After HC is trained, the output class for an input vector *att* is found using the following formula $Cl(att) = \arg\max_{i \in \mathcal{K}} Cl(i|att)$, where

$$Cl(i|att) = \sum_{j: i \in C_j} Cl_{mod}(C_j|att) Cl_j(i|att) \tag{1}$$

and $Cl_{mod}(C_j|att)$ stands for the probability of the selection of cluster C_j, provided that the vector *att* is given into the classifier's input. Cl_j is a classifier associated with cluster C_j (that is, it is able to recognize only classes that belong to C_j). By $Cl_j(i|att)$ we mean the activation of Cl_j for i-th class, given *att* vector as the classifier's input. If a given classifier Cl does not divide its problem into subproblems (i.e. Cl is a leaf in the classifiers tree), its answer is just a probability vector: $Cl(att) = [p(1) ... p(K)]$.

A two level HC is depicted in Figure 1. The crucial part of the HC construction is the clustering process. It is important to note, that the clusters in HC may *overlap*, *i.e.* $\exists\ i, j, i \neq j : C_i \cap C_j \neq \emptyset$, none is a subset of another cluster, and none is composed of all classes from $\mathcal{K}$. Thus, it can be shown that addition of classifier layer results in low error.

When studying the properties of HC, it became apparent that the overall accuracy depends on the clustering found in the algorithm, reflected in the correct value found with Cl_{mod}. The actual clusterings are found using machine learning approaches [4, 3]. We have noted that the actual number of possible clusterings is not known, and it became the motivation for this work.

Moreover, the clustering counting problem itself has not been pursued before and can be treated as a purely mathematical problem. The cluster number sequences we have obtained are not to be found in the integer sequence database [5].

In the following sections we shall define the problem of finding the number of clusters formally and attempt to find an exact formula for a small number of clusters. The algorithm for explicit selection of all possible clusterings will be shown.

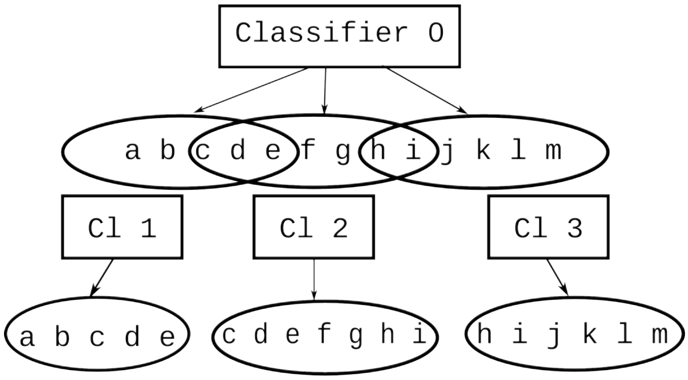

Fig. 1. The Hierarchical Classifier HC

2. Problem formulation

Let us formulate the clusterings counting task in a formal way. Fix $K, J \in \mathbb{N}$ such that $K > J \geq 2$. Let $\mathcal{K} = \{x_1, \ldots, x_K\}$ be the set of all classes. We need to define a few types of families of sets.

DEFINITION 2. *A family* $\mathfrak{C} = \{C_1, C_2, \ldots, C_J\}$ *of* J *sets is the proper* (K, J)*-clustering of* $\mathcal{K}$ *iff the following conditions are fulfilled:*

$$\forall i = 1, 2, \ldots, J \; C_i \subsetneq \mathcal{K} = \{x_1, \ldots, x_K\}, \tag{2}$$

$$\bigcup \mathfrak{C} = \mathcal{K}, \tag{3}$$

$$\forall i = 1, 2, \ldots, J \; |C_i| \geq 2, \tag{4}$$

$$\forall i, j = 1, 2, \ldots, J \; C_i \subseteq C_j \Rightarrow i = j, \tag{5}$$

$$\exists i, j \in \{1, \ldots, J\}, \; i \neq j : \; C_i \cap C_j \neq \emptyset. \tag{6}$$

If a family $\mathfrak{C}$ fulfills (2),(3),(5),(6) and does not fulfill (4), we will call $\mathfrak{C}$ the improper (K, J)-clustering. If $\mathfrak{C}$ fulfills (2),(3) and does not fulfill (5), we will call it the (K, J)-clustering with inclusion. By a (K, J)-family over $\mathcal{K}$ we understand any family of J sets fulfilling conditions (2) and (3).

Our main problem is, given $K > J \geq 2$, to compute the number of all proper (K, J)-clusterings. We denote this number by $\vartheta(K, J)$:

$$\vartheta(K, J) = |\{\mathfrak{C} : \mathfrak{C} \text{ is a proper } (K, J)\text{-clustering}\}| \,. \tag{7}$$

We introduce the linear order $<_{\mathcal{K}}$ on $\mathcal{K}$ and assume that if $\mathcal{K} = \{x_1, \ldots, x_K\}$, then $x_1 <_{\mathcal{K}} x_2 <_{\mathcal{K}} \cdots <_{\mathcal{K}} x_K$. If $Y \subseteq \mathcal{K}$, then by $\max(Y)$ we denote the maximal element in Y, with respect to relation $<_{\mathcal{K}}$.

3. Case $J = 2$

The case for $J = 2$ is easy to solve:

THEOREM 3.
$$\vartheta(K,2) = 3S(K,3), \tag{8}$$
where $S(K,3)$ is the Stirling number of the second kind.

Proof. Consider any partition of K-element set into 3 nonempty subsets C_1, C_2, C_3. Such a partition defines three different proper $(K,2)$-clusterings: $\mathfrak{C}_1 = \{C_1 \cup C_2, C_1 \cup C_3\}$, $\mathfrak{C}_2 = \{C_1 \cup C_2, C_2 \cup C_3\}$, and $\mathfrak{C}_3 = \{C_1 \cup C_3, C_2 \cup C_3\}$. On the other hand, each proper $(K,2)$-clustering can be identified with some partition of $\mathcal{K}$ into 3 nonempty subsets with one of the subsets marked, representing the intersection of clusters. □

In the following example we enumerate all 18 proper $(4,2)$-clusterings.

EXAMPLE 1. $\vartheta(4,2) = 18$.

{{1,2},{1,3,4}}	{{1,2},{2,3,4}}	{{1,3},{1,2,4}}	{{1,3},{2,3,4}}
{{1,4},{1,2,3}	{{1,4},{2,3,4}}	{{2,3},{1,2,4}}	{{2,3},{1,3,4}}
{{2,4},{1,2,3}}	{{2,4},{1,3,4}}	{{3,4},{1,2,3}}	{{3,4},{1,2,4}}
{{2,3,4},{1,3,4}}	{{2,3,4},{1,2,4}}	{{2,3,4},{1,2,3}}	{{1,3,4},{1,2,4}}
{{1,3,4},{1,2,3}}	{{1,2,4},{1,2,3}}		

For example, partition $\{\{1\},\{2,3\},\{4\}\}$ defines three different $(4,2)$-clusterings, namely $\{\{1,2,3\},\{1,4\}\}$, $\{\{1,2,3\},\{2,3,4\}\}$ and $\{\{1,4\},\{2,3,4\}\}$.

The sequence $\{\vartheta(K,2)\}_{K=0}^{\infty} = (0,0,0,3,18,75,270,903,\dots)$ is known as the "number of connected 2-element antichains on a labeled n-set" [5]. For $J = 3$ we have not found a similar sequence; therefore the former one can be considered as a special case (for $J = 2$) of the family $\{\vartheta(K,J)\}_{K=0}^{\infty}$ of sequences.

4. Analysis of the general case

Before we pass to the case of $J = 3$, we shall analyze the general case. The observations done in this section will be useful in constructing the recurrence relation for $J = 3$.

We would like to express $\vartheta(K+1,J)$ in terms of $\vartheta(L,J)$ for $L \le K$, or in terms of other formulae given *explicite*.

DEFINITION 4. *Let $\mathfrak{C} = \{C_1, C_2, \dots, C_J\}$ be a family of subsets such that $\bigcup \mathfrak{C} = \{x_1, x_2, \dots, x_K\}$. If there exists $I \subseteq \{1,2,\dots,J\}$ such that $\mathfrak{C}' = \bigcup_{i \in I}\{C_i \cup \{x_{K+1}\}\} \cup \bigcup_{j \in \{1,\dots,J\}\setminus I}\{C_j\}$ is a proper $(K+1,J)$-clustering, then $\mathfrak{C}$ will be called an extendable family of sets.*

PROPOSITION 5. *Let $\mathfrak{C}$ and $\mathfrak{C}'$ be the families from Definition 4. If x_{K+1} belongs to exactly one subset of $\mathfrak{C}'$, then $\sum_{i=1}^{J}|C_i| > K$.*

Proof. Contrarily, suppose $\sum_{i=1}^{J}|C_i| = K$. Then $\mathfrak{C}$ is a partition of $\mathcal{K}$. But x_{K+1} belongs only to one element of $\mathfrak{C}'$, so $\mathfrak{C}'$ is also a partition – a contradiction with $(K+1, J)$-clustering of $\mathfrak{C}'$. □

PROPOSITION 6. *If $\mathfrak{C} = \{C_1, \dots, C_J\}$ is an extendable family, then all C_i's are different.*

Proof. Let $\mathfrak{C}'$ be a family formed from $\mathfrak{C}$ by adding a new element to some elements of $\mathfrak{C}$. Suppose contrarily that there exist $i, j, i \neq j$, such that $C_i = C_j$. But then, after adding a new x_{K+1} element, in $\mathfrak{C}'$ there exist two different elements D_1, D_2 such that $D_1 = C_1$ or $D_1 = C_1 \cup \{x_{K+1}\}$ and $D_2 = C_2$ or $D_2 = C_2 \cup \{x_{K+1}\}$. In each of these four situations either $D_1 = D_2$ or $D_1 \subset D_2$ or $D_2 \subset D_1$ – a contradiction with $(K+1, J)$-clustering of $\mathfrak{C}'$. □

A proper $(K+1, J)$-clustering can be build by adding a new x_{K+1} element to some subsets of an extendable (K, J)-family over $\mathcal{K} = \{x_1, \dots, x_K\}$. For some extendable (K, J)-families over $\mathcal{K}$, adding a new element can be done in more than one way.

Lemma 7 characterizes the extendable (K, J)-families.

LEMMA 7. *Let $\mathfrak{C} = \{C_1, \dots, C_J\}$ be an extendable (K, J)-family over $\mathcal{K} = \{x_1, \dots, x_K\}$. Then $\mathfrak{C}$ has exactly one of the following four properties:*

(P1) $\mathfrak{C}$ is a proper (K, J)-clustering;

(P2) $\mathfrak{C}$ is an improper (K, J)-clustering;

(P3) $\mathfrak{C}$ is a partition of K-element set into J nonempty subsets;

(P4) $\mathfrak{C}$ is a (K, J)-clustering with inclusion, such that $C_i \subseteq C_j \subseteq C_k \Rightarrow i = j \vee j = k$.

Proof. It is clear that $\mathfrak{C}$ cannot have two or more properties (P1)–(P4). Let $\mathcal{D} = \{D_1, \dots, D_J\}$ be a proper $(K+1, J)$-clustering. We will show that removing $\max(\bigcup \mathcal{D}) = x_{K+1}$ from all subsets of $\mathcal{D}$ gives us a family with one of the properties (P1)–(P4). Let $\delta = |\{D_i : x_{K+1} \in D_i\}|$, $\varepsilon = \max_{j \in \{1,\dots,J\}}\{|\{D_i : j \in D_i\}|\}$. It is clear that both $\delta \geq 1$ and $\varepsilon \geq 1$. Consider three possible cases:

Case 1. $\delta = 1$. From Proposition 5 $\varepsilon > 1$, so in $\mathfrak{C}$ there exist C_1, C_2, $C_1 \neq C_2$ such that $C_1 \cap C_2 \neq \emptyset$. If $C_1 \subset C_2$, then from Proposition 6 it is a strict inclusion and because $\delta = 1$, no two other subsets remain in the inclusion relation. In this situation $\mathfrak{C}$ has property (P4). If no two subsets are in the inclusion relation, then $\mathfrak{C}$ has property (P1) or (P2).

Case 2. $\varepsilon = 1$. Then $\mathfrak{C}$ must be a partition and therefore has property (P3).

Case 3. $\delta > 1, \varepsilon > 1$. The case reduces to Case 1 or there is more than one pair of sets C_i, C_j such that $C_i \subset C_j$. Such a family cannot have properties (P1)–(P3). Because of extendability, $\mathfrak{C}$ must have the property (P4). □

Lemma 7 gives us the complete list of ways in which a proper $(K+1, J)$-clustering can be formed. The only thing to do is to count the number of extendable families with properties (P1)–(P4) and, for each of them, the number of ways in which a new x_{K+1}th element can be added to form a proper $(K+1, J)$-clustering.

5. Case $J = 3$

We will use Lemma 7 to count the number of ways to extend each extendable family into a proper $(K+1, 3)$-clustering.

LEMMA 8. *Let $K \geq 3$. There exist $7\vartheta(K, 3)$ proper $(K+1, 3)$-clusterings created by extending some $(K, 3)$-family with property (P1).*

Proof. Having a proper $(K, 3)$-clustering $\mathfrak{C} = \{C_1, C_2, C_3\}$ we must add x_{K+1} at least to one of C_i's and at most to all of them. So there are 7 ways to do that. □

LEMMA 9. *Let $K \geq 3$. There exist $\frac{2K}{3}3^K - 3K \cdot 2^K + 6K$ proper $(K+1, 3)$-clusterings created by extending some $(K, 3)$-family with property (P2).*

Proof. Assume first $K \geq 4$. For a $(K, 3)$-family $\mathfrak{C} = \{C_1, C_2, C_3\}$, in order to have property (P2), exactly one of the C_i's must be a singleton. First let us count the number of extendable families with property (P2). Let $\{C_1, C_2, C_3\}$ be such a family and let $|C_1| = 1$. We can choose C_1 in K different ways and two other sets must constitute a proper $(K-1, 2)$-clustering, so there are $K\vartheta(K-1, 2)$ such families. For each of them x_{K+1} must be added to C_1 in order to preserve (4). There are 4 possibilities to add x_{K+1} to C_2 and C_3: $x_{K+1} \in C_2 \setminus C_3$, $x_{K+1} \in C_3 \setminus C_2$, $x_{K+1} \in C_2 \cap C_3$, $x_{K+1} \notin C_2 \cup C_3$. The number of desired clusterings is therefore

$$4K\vartheta(K-1, 2) = 4K \sum_{i=1}^{K-3} \binom{K-1}{i} \cdot (2^{K-i-1} - 1) = \frac{2K}{3}3^K - 3K \cdot 2^K + 6K.$$

Note, that if the family has property (P2), then necessarily $K \geq 4$, but for $K = 3$ the formula is still valid, because $4 \cdot 3 \cdot \vartheta(2, 2) = 0$. □

LEMMA 10. *Let $K \geq 3$. There exist $\frac{2}{3}3^K - \left(\frac{K}{4} + 2\right) \cdot 2^K + K + 2$ proper $(K+1, 3)$-clusterings created by extending some $(K, 3)$-family with property (P3).*

Proof. Let $\{C_1, C_2, C_3\}$ be a partition of $\mathcal{K} = \{x_1, \ldots, x_K\}$. Consider 3 possible cases.

Case 1. $|C_1| = 1$, $|C_2| = 1$, $|C_3| > 1$. Let $K \geq 4$. Two singleton sets, C_1 and C_2 can be chosen in $\alpha = \binom{K}{2}$ ways and C_3 is uniquely determined. x_{K+1} must be added to C_1 and C_2 in order to preserve (4). It can be added or not to C_3, so eventually we have $2\binom{K}{2}$ ways to create a proper $(K+1, 3)$-clustering.

Case 2. $|C_1| = 1$, $|C_2| > 1$, $|C_3| > 1$. Let $K \geq 5$. C_1 can be chosen in K ways. C_2 and C_3 form a partition of a $K-1$-element set. C_2 can be chosen in $\sum_{i=2}^{K-3} \binom{K-1}{i} = 2^{K-1} - 2K$ ways, then C_3 is uniquely determined. Because the order of C_2 and C_3 is not important, we must divide the value by 2, so we have $\beta = \frac{K}{2}(2^{K-1} - 2K)$ partitions with exactly one singleton. For each such an extendable family x_{K+1} can be added in 3 ways: it must be added to C_1 and it must be added to at least one of the sets C_2, C_3. Finally, we obtain $\frac{3K}{2}(2^{K-1} - 2K) = \frac{3K}{4}2^K - 3K^2$ ways to create a proper $(K+1, 3)$-clustering.

Case 3. $|C_i| > 1$ $\forall i = 1, 2, 3$. Let $K \geq 6$. For $K \geq 6$ there are $S(K, 3) - \alpha - \beta$ such partitions. For each of them, x_{K+1} must be added to at least two subsets, so we have 4 possibilities of doing this and $4(S(K, 3) - \alpha - \beta) = \frac{2}{3} \cdot 3^K - (K + 2) \cdot 2^K + 2K^2 + 2K + 2$ proper $(K+1, J)$-clusterings.

Getting all three cases together we obtain the desired number of clusterings for $K \geq 6$. Notice that the formula in Case 2. is still valid for $K = 4$ because it equals $\frac{3 \cdot 4 \cdot 2^2}{4} - 3 \cdot 4^2 = 0$. Also the formula in Case 3. is still valid for $K = 4$ or $K = 5$. In both cases it equals 0. The formula from the thesis is also valid for $K = 3$ and it equals 1: in order to form a proper $(4, 3)$-clustering from the partition of 3-element set into 3 singletons we need to add x_{K+1} to all three sets and we can do it only in one way. Therefore the thesis holds for all $K \geq 3$. $\square$

LEMMA 11. *Let* $K \geq 3$. *There exist* $2 \cdot 5^K - 3 \cdot 4^K - \frac{5}{2} \cdot 3^K + \left(6 - \frac{K}{2}\right) \cdot 2^K + 2K - \frac{5}{2}$ *proper* $(K+1, 3)$*-clusterings created by extending some* $(K, 3)$*-family with property (P4).*

Proof. Each extendable $(K, 3)$-family $\mathfrak{C} = \{C_1, C_2, C_3\}$ with property (P4) and such that $C_1 \subset C_2$ can be uniquely represented by a 'multi-characteristic' vector $p(\mathfrak{C}) = (p_1, p_2, \ldots, p_K)$ in which $p_i \in \{A, B, C, D, E\}$ represents the position of x_i in $\mathfrak{C}$ in the following way:

$$\begin{aligned} p_i = A &\quad \Leftrightarrow \quad x_i \in C_1 \cap C_2 \cap C_3, \\ p_i = B &\quad \Leftrightarrow \quad x_i \in C_2 \cap C_3 \land x_i \notin C_1, \\ p_i = C &\quad \Leftrightarrow \quad x_i \in C_1 \cap C_2 \land x_i \notin C_3, \\ p_i = D &\quad \Leftrightarrow \quad x_i \in C_3 \land x_3 \notin C_1 \cup C_2, \\ p_i = E &\quad \Leftrightarrow \quad x_i \in C_2 \land x_3 \notin C_1 \cup C_3. \end{aligned}$$

Note, that no other possibility is allowed, because $C_1 \subset C_2$ implies that if $x_i \in C_1$ then necessarily $x_i \in C_2$ and thus $x_i \in C_1 \cup C_2 \cup C_3$ for all $i = 1, 2, \ldots, K$. It is clear that there is a well defined bijection between the set of all 'multi-characteristic' vectors $\{A, B, C, D, E\}^K$ and the set of all $(K, 3)$-families (C_1, C_2, C_3) with all C_i's ordered, and such that $C_1 \subseteq C_2$. We will count the number of all vectors representing different extendable $(K, 3)$-families with property (P4). For the sake of simplicity, for a given vector p we will use A (resp. B,C,D,E) for denoting the number of $p_i = A$ (resp. B, C, D, E) in p. This will not lead to any misunderstanding. Naturally, for each p we have $A + B + C + D + E = K$. For a $(K, 3)$-family to be an extendable one with property (P4) some conditions must be fulfilled and these conditions can be transformed into some algebraic relations concerning A, B, C, D, E values. The set of all conditions and the corresponding algebraic relations are given below.

$$\begin{aligned} C_1 \subsetneq C_2 &\Rightarrow B+E>0 && (9)\\ |C_1| \geq 1 &\Rightarrow A+C>0 && (10)\\ C_3 \not\subset C_1 &\Rightarrow B+D>0 && (11)\\ C_2 \not\subset C_3 &\Rightarrow C+E>0 && (12)\\ \neg(C_1 \subseteq C_3 \subseteq C_2) &\Rightarrow C+D>0. && (13) \end{aligned}$$

Conditions (9) and (10) come directly from the assumptions on the $(K,3)$-family. Conditions (11)–(13) come from the property (P4) which says that 3 different sets in $\mathfrak{C}$ cannot form a descending family (w.r.t. the inclusion). We will count the number of vectors fulfilling (9)–(13). There are 12 possible cases (see Fig. 2):

(BC) $C_3 = C_2 \setminus C_1$,

(ABC) $C_2 = C_1 \cup C_3$, $C_1 \cap C_3 \neq \emptyset$,

(ADE) $C_2 \setminus C_3 \neq \emptyset$, $C_3 \setminus C_2 \neq \emptyset$, $C_2 \cap C_3 = C_1$,

(BCD) $C_2 \subsetneq C_1 \cup C_3$, $C_1 \cap C_3 = \emptyset$,

(BCE) $C_1 \cap C_3 = \emptyset$, $C_1 \cup C_3 \subsetneq C_2$,

(CDE) $C_3 \cap C_2 = \emptyset$,

(ABCD) $C_3 \cap C_1 \neq \emptyset$, $C_1 \setminus C_3 \neq \emptyset$, $C_2 \setminus C_1 \subsetneq C_3$, $C_3 \setminus C_2 \neq \emptyset$,

(ABCE) $C_3 \cap C_1 \neq \emptyset$, $C_3 \subsetneq C_2$,

(ABDE) $C_1 \subsetneq C_3$, $C_3 \cap (C_2 \setminus C_1) \neq \emptyset$, $C_3 \setminus C_2 \neq \emptyset$,

(ACDE) $C_3 \cap C_1 \neq \emptyset$, $C_1 \setminus C_3 \neq \emptyset$, $C_3 \cap (C_2 \setminus C_1) = \emptyset$, $C_3 \setminus C_2 \neq \emptyset$,

(BCDE) $C_3 \cap C_1 = \emptyset$, $C_3 \cap C_2 \neq \emptyset$, $C_3 \setminus C_2 \neq \emptyset$, $C_2 \setminus (C_1 \cup C_3) \neq \emptyset$,

(ABCDE) $C_3 \cap C_1 \neq \emptyset$, $C_1 \setminus C_3 \neq \emptyset$, $C_3 \cap (C_2 \setminus C_1) \neq \emptyset$, $C_2 \setminus (C_1 \cup C_3) \neq \emptyset$, $C_3 \setminus C_2 \neq \emptyset$.

If vector p falls under the case (X), we will say that p is of type (X). The case number symbolically describe the set of allowed values in p falling under that case. For example, if p is of type (BC), then p contains only B's and C's and $B, C > 0$. Note, that in some cases two different vectors can represent the same $(K,3)$-family. This case holds if there is a "symmetry" between C_1 and C_3 or between C_2 and C_3. For example, vectors (A,B,C) and (A,C,B) represent one family in two ways: in the first $x_2 \in C_3, x_3 \in C_1$ and in the second $x_2 \in C_1, x_3 \in C_3$. After changing C_1 with C_3 the structure of the family remains the same. Such a symmetry occurs in cases (BC), (ABC), (ADE), (BCE), (ABCE) and (ABDE), so in each of these cases the number of all different vectors of a given type must be divided by 2. Let $\alpha(X)$ denote the number of proper $(K+1,3)$-clusterings created from $(K,3)$-family of type (X). Now we will count $\alpha(X)$ for all 12 cases. In all of them, in order to create a proper $(K+1,3)$-clustering, a new element x_{K+1} cannot be added to C_2

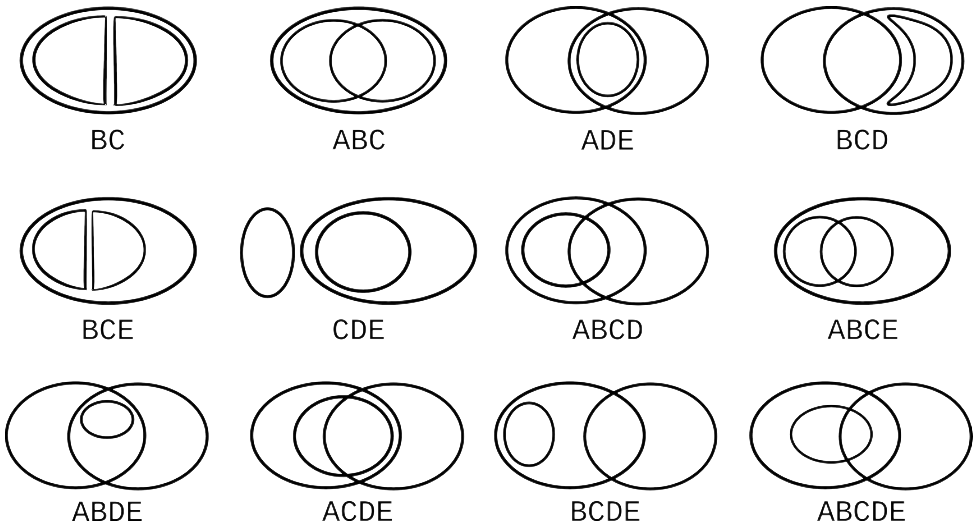

Fig. 2. All possible cases

and must be added to C_1, because of inclusion $C_1 \subsetneq C_2$. Therefore, the number of ways of creating a proper $(K+1,3)$-clustering from a given family depends only on the fact whether we *have to* add x_{K+1} to C_3 or we *may* do this. In the first case there is only one way to create a proper clustering; in the second one there are two ways of doing that.

Case (BC). There are $\frac{2^K-2}{2}$ different $(K,3)$-families of type (BC). In order to extend it into a proper $(K+1,3)$-family we must add x_{K+1} to C_3; therefore we can do it in only one way, so $\alpha(BC) = 1 \cdot \frac{2^K-2}{2}$.

Cases (ABC), (ADE) and (BCE). These cases are identical modulo type names. From the inclusion-exclusion principle we have that there are $3^K - \binom{3}{2}2^K + \binom{3}{1}$ vectors of type (ABC). In cases (ABC) and (BCE) x_{K+1} must be added to C_3; in (ADE) it cannot be added, therefore in each of these cases there is only one possibility of adding x_{K+1}. Because of the symmetry between C_1 and C_3 in (ABC) and (BCE) and between C_2 and C_3 in (ADE) we have $\alpha(ABC) = \alpha(ADE) = \alpha(BCE) = 1 \cdot \frac{3^K - \binom{3}{2}2^K + \binom{3}{1}}{2}$.

Case (BCD). x_{K+1} can be added or not to C_3; the case is not a symmetric one, so $\alpha(BCD) = 2\alpha(ADE)$.

Case (CDE). This is a nonsymmetric case. Consider two subcases: $|C_3| = 1$ and $|C_3| > 1$. If $|C_3| = 1$, then we must add x_{K+1} to C_3 and there are $1 \cdot K(2^{K-1} - 2)$ vectors of this type. If $|C_3| > 1$ then x_{K+1} can be added or not to C_3 and there are $\sum_{i=2}^{K-2} \binom{K}{i}(2^{K-i}-2)$ vectors of this type (here i goes through the number of elements in C_3). Eventually, we have $\alpha(CDE) = K(2^{K-1} - 2) + 2 \cdot \sum_{i=2}^{K-2} \binom{K}{i}(2^{K-i} - 2) = 2 \cdot 3^K - (\frac{K}{2} + 6) \cdot 2^K + 2K + 6$.

Cases (ABCD), (ACDE) and (BCDE). These are nonsymmetric cases, identical modulo type names, and x_{K+1} can be added or not to C_3. We have $\alpha(ABCD) = \alpha(ACDE) = \alpha(BCDE) = 2(4^K - \binom{4}{3}3^K + \binom{4}{2}2^K - \binom{4}{1})$.

Cases (ABCE) and (ABDE). We have a symmetry between C_1 and C_3 in (ABCD) and between C_2 and C_3 in (ABDE). The new x_{K+1} element must be added to C_3 in (ABCE) and cannot be added to C_3 in (ABDE). In both cases there is only one way to create a proper clustering and $\alpha(ABCE) = \alpha(ABDE) = \frac{4^K - \binom{4}{3}3^K + \binom{4}{2}2^K - \binom{4}{1}}{2}$.

Case (ABCDE). In this nonsymmetric case x_{K+1} can be added or not to C_3, so $\alpha(ABCDE) = 2(5^K - \binom{5}{4}4^K + \binom{5}{3}3^K + \binom{5}{2}2^K - \binom{5}{1})$.

Let I be the set of all 12 possible types. Getting all the cases together we obtain

$$\sum_{X \in I} \alpha(X) = 2 \cdot 5^K - 3 \cdot 4^K - \frac{5}{2} \cdot 3^K + \left(6 - \frac{K}{2}\right) \cdot 2^K + 2K - \frac{5}{2},$$

and notice that the formula is also valid for $K = 4, 5$, therefore it is valid for each $K \geq 4$. This ends the proof of Lemma 10. □

Now we are ready to state the main theorem.

THEOREM 12.

$$\begin{aligned} \vartheta(3,3) &= 1 \\ \vartheta(K+1,3) &= 7\vartheta(K,3) + 2 \cdot 5^K - 3 \cdot 4^K + \frac{4K-11}{6} \cdot 3^K + (4 - \frac{15K}{4}) \cdot 2^K \\ &\quad +9K - \frac{1}{2}, \end{aligned}$$

where $K \geq 3$.

Proof. The proof comes directly by applying Lemmata 7 and 8–11. □

In Tab. 1 some values of the $\vartheta(K,3)$ are given. It can be noticed, that the growth rate is exponential and is $O(n^K)$ where n is the number of all properties of elements of multi-characteristic vector $p(\mathfrak{C})$ (see Lemma 11). It can be easily shown that for arbitrary $J \geq 3$ value $n = 2^{J-1} + 2^{J-2} + 1$. In other words, the exponent in the growth rate is the multiple of number of clusters J and number of classes K.

Tab. 1. Some first values of $\vartheta(K,3)$.

K	3	4	5	6	7	8	9	10
$\vartheta(K,3)$	1	38	675	7 840	74 291	630 546	5 014 843	38 290 580

6. Algorithm for clusterings generation for $J = 3$

Generation of all possible $(K,3)$-clusterings can be done in several ways. The simplest one is to generate all 0-1 matrices $M^{K\times 3}$, representing clusterings (that is,

$M_{i,j} = 1 \Leftrightarrow x_i \in C_j$) and for each matrix check if it fulfills all assumptions for clustering to be proper. But basing on Lemma 7 and the proof of Theorem 12 we can construct an algorithm which produces all possible $(K, 3)$-clusterings and nothing more. This algorithm is shown in listing 4. It uses four procedures, corresponding to Lemmata 8–11. The procedures are shown in listings 1–3. The fourth procedure is described in an informal way, in order to simplify the considerations.

Algorithm 1 GenProperClusterings

INPUT: $X = \{x_1, \ldots, x_K\}$
OUTPUT: Q – the set of all $(K, 3)$-proper clusterings created from extendable $(K-1, 3)$ families with property (P1).
if $K < 3$ **then return** $\emptyset$;
if $K == 3$ **then return** $\{\{x_1, x_2\}, \{x_1, x_3\}, \{x_2, x_3\}\}$
else
  $P =$GenProperClusterings$(K-1, X \setminus \{x_K\})$;
  $Q = \emptyset$;
  foreach $p = \{C_p^1, C_p^2, C_p^3\} \in P$ **do**
    $Q = Q \cup \{\{C_p^1 \cup \{x_K\}, C_p^2, C_p^3\}\} \cup \{\{C_p^1, C_p^2 \cup \{x_K\}, C_p^3\}\}$;
    $Q = Q \cup \{\{C_p^1, C_p^2, C_p^3 \cup \{x_K\}\}\} \cup \{\{C_p^1 \cup \{x_K\}, C_p^2 \cup \{x_K\}, C_p^3\}\}$;
    $Q = Q \cup \{\{C_p^1 \cup \{x_K\}, C_p^2, C_p^3 \cup \{x_K\}\}\} \cup \{\{C_p^1, C_p^2 \cup \{x_K\}, C_p^3 \cup \{x_K\}\}\}$;
    $Q = Q \cup \{\{C_p^1 \cup \{x_K\}, C_p^2 \cup \{x_K\}, C_p^3 \cup \{x_K\}\}\}$;
  return Q;

Algorithm 2 GenImproperClusterings

INPUT: $X = \{x_1, \ldots, x_K\}$
OUTPUT: Q – the set of all $(K, 3)$-proper clusterings created from extendable $(K-1, 3)$ families with property (P2).
if $K \leq 4$ **then return** $\emptyset$
else
  $Q = \emptyset$;
  for $i = 1$ **to** $K - 1$
    $R =$Gen2ProperClustering$(X \setminus \{x_i, x_K\})$;
    foreach $r = \{C_r^1, C_r^2\} \in R$ **do**
      $Q = Q \cup \{\{C_r^1 \cup \{x_K\}, C_r^2, \{x_i, x_K\}\}\} \cup \{\{C_r^1, C_r^2 \cup \{x_K\}, \{x_i, x_K\}\}\}$;
      $Q = Q \cup \{\{C_r^1, C_r^2, \{x_i, x_K\}\}\} \cup \{\{C_r^1 \cup \{x_K\}, C_r^2 \cup \{x_K\}, \{x_i, x_K\}\}\}$;
  return Q;

Procedure Gen2ProperClustering(X) generates all proper $(K, 2)$-clusterings. It can be simply implemented in a following way: for a given X generate all $Y \subset X$ such that $1 \leq |Y| \leq |X| - 2$, then generate all partitions $\{P, R\}$ of $X \subset Y$ for 2 subsets and put $C_1 = P \cup Y$, $C_2 = R \cup Y$. Algorithms for generating subsets and partitions of sets are well-known (see for example [6]).

The idea of the GenInclusionClusterings is similar to the one from algorithms 1–3. As the input the algorithm receives $\{x_1, \ldots, x_K\}$ and generates all proper $(K, 3)$-clusterings from extendable $(K-1, 3)$ families with property (P4).

Algorithm 3 GENPARTITIONCLUSTERINGS

INPUT: $X = \{x_1, \ldots, x_K\}$
OUTPUT: Q – the set of all $(K, 3)$-proper clusterings created from extendable $(K-1, 3)$ families with property (P3).
if $K < 3$ **then return** $\emptyset$
else
  $Q = \emptyset$;
  foreach $p = \{C_1, C_2, C_3\}$ – partition of $X \setminus \{x_K\}$, such that $|C_1| \leq |C_2| \leq |C_3|$
    if $|C_1| = |C_2| = |C_3| = 1$ **then**
      return $\{\{x_1, x_4\}, \{x_2, x_4\}, \{x_3, x_4\}\}$;
    if $|C_1| = |C_2| = 1, |C_3| > 1$ **then return**
      $\{\{C_1 \cup \{x_K\}, C_2 \cup \{x_K\}, C_3\}\} \cup \{\{C_1 \cup \{x_K\}, C_2 \cup \{x_K\}, C_3 \cup \{x_K\}\}\}$;
    if $|C_1| = 1, |C_2| > 1$ **then**
      return $\{\{C_1 \cup \{x_K\}, C_2 \cup \{x_K\}, C_3\}\} \cup \{\{C_1 \cup \{x_K\}, C_2, C_3 \cup \{x_K\}\}\} \cup$
      $\cup \{\{C_1 \cup \{x_K\}, C_2 \cup \{x_K\}, C_3 \cup \{x_K\}\}\}$;
    if $|C_1| > 1$ **then**
      return $\{\{C_1 \cup \{x_K\}, C_2 \cup \{x_K\}, C_3\}\} \cup \{\{C_1 \cup \{x_K\}, C_2, C_3 \cup \{x_K\} \cup$
      $\cup \{\{C_1, C_2 \cup \{x_K\}, C_3 \cup \{x_K\}\}\} \cup \{\{C_1 \cup \{x_K\}, C_2 \cup \{x_K\}, C_3 \cup \{x_K\}\}\}$;

For each of the 12 cases (BC), (ABC), (ADE),. . . , (ABCDE), defined in the proof of Lemma 11, a set S of $K-1$-element vectors is generated. Vector corresponding to a given case contains at least one of each property defined in this case and does not contain any other properties. In this stage symmetries are excluded; for example, in case (BC) vectors (B,B,C,B,C) and (C,C,B,C,B) represent the same $(5, 3)$-clustering with inclusion. Therefore one of this vectors is excluded from S. For each case we know what are the possibilities of adding a new element x_K. We perform all allowed adding operations for all vectors in S. For example, for vector (BBCBC) we know that because it is of type (BC), x_K must be added to C_1 and C_3, and cannot be added to C_2 – we have only one way to add x_K and we obtain a proper $(K, 3)$-clustering $\{C_1 \cup \{x_K\}, C_2, C_3 \cup \{x_K\}\}$. The other cases are dealt with in a similar way.

Algorithm 4 GENALLCLUSTERINGS

INPUT: $X = \{x_1, \ldots, x_K\}$
OUTPUT: Q – the set of all $(K, 3)$-proper clusterings
  $Q = \emptyset$;
  $Q = Q \cup$GENPROPERCLUSTERINGS(X);
  $Q = Q \cup$GENIMPROPERCLUSTERINGS(X);
  $Q = Q \cup$GENPARTITIONCLUSTERINGS(X);
  $Q = Q \cup$GENINCLUSIONCLUSTERINGS(X);
  return Q;

7. Dedekind problem

The Dedekind problem concerns determining exact formula for the number of monotonic boolean functions with fixed number of variables. These numbers – known as Dedekind numbers – form a rapidly growing sequence (denoted by $\psi(n)$ where n is the number of function variables) and also define the numbers of antichains of n-element set.

Let us introduce the necessary definitions.

Definition 13. ***A partially ordered set*** *is a pair* $P = (X, \sqsubseteq)$ *where* X *is a set and* $\sqsubseteq$ *is a binary relation over* X *which fulfils following conditions:*

1. $\forall_{x \in X} \quad x \sqsubseteq x$ *(reflexivity),*
2. $\forall_{x,y \in X} \quad x \sqsubseteq y \wedge y \sqsubseteq x \Rightarrow x = y$ *(antisymmetry),*
3. $\forall_{x,y,z \in X} \quad x \sqsubseteq y \wedge y \sqsubseteq z \Rightarrow x \sqsubseteq z$ *(transitivity).*

An example of partially ordered set is a power set with inclusion relation.

Definition 14. ***A chain*** *in a partially ordered set* $P = (X, \sqsubseteq)$ *is a subset* A *of set* X *whose any pair of elements are comparable, i.e.*

$$\forall_{x,y \in A} \quad x \sqsubseteq y \quad \vee \quad y \sqsubseteq x.$$

Definition 15. ***An antichain*** *in a partially ordered set* $P = (X, \sqsubseteq)$ *is a subset* A *of set* X *in which any two elements are not comparable, i.e.*

$$\forall_{x,y \in A} \quad x \not\sqsubseteq y \quad \wedge \quad y \not\sqsubseteq x.$$

Definition 16. ***A boolean function*** *is a function* $f : X \to Y$*, where* $X \subset \{0,1\}^n$ *and* $Y \subset \{0,1\}$*. A boolean function is called* ***monotonic*** *if*

$$\forall_{a_1,\ldots,a_n,b_1,\ldots,b_n \in \{0,1\}} \quad a_1 \leqslant b_1, \ldots, a_n \leqslant b_n \Rightarrow f(a_1, \ldots, a_n) \leqslant f(b_1, \ldots, b_n).$$

Monotonic boolean functions are an important class of boolean functions. Their characteristic is that they can be defined by composition of logical conjunctions and disjunctions, but not negations.

The problem of determining $\psi(n)$ was formulated in 1897 by Richard Dedekind [7]. He solved it for values $n \leqslant 4$. In 1940 Church [8] presented the solution for $n = 5$, whereas Ward [9] for $n = 6$.

More general properties were proved later. In 1953 Yamamoto [10] showed that $\psi(n)$ is even for even values of n. In 1954 Gilbert [11] proved the inequality

$$2^{\binom{n}{[n/2]}} \leqslant \psi(n) \leqslant n^{\binom{n}{[n/2]}+2},$$

while Yamamoto [12]

$$\log_2 \psi(n) < \binom{n}{[n/2]} \left(1 + \mathcal{O}\left(n^{-1}\right)\right) \log_2 \sqrt{\frac{\pi n}{2}}.$$

In [13] Korobkov improved the upper bound of $\psi(n)$ for $2^{4.23\binom{n}{[n/2]}}$. In 1966 Hansel [14] managed to move it to $3^{\binom{n}{[n/2]}}$.

In Kleitman's paper [15] it is shown that

$$2^{(1+\alpha_n)\binom{n}{[n/2]}} \leqslant \psi(n) \leqslant 2^{(1+\beta_n)\binom{n}{[n/2]}},$$

where $\alpha_n = ce^{-\frac{n}{4}}$, $\beta_n = c'(\log n)/n^{\frac{1}{2}}$.

The more precise estimation was reached by Korshunov [16]:

$$\psi(n) \sim 2^{\binom{n}{[n/2]}} \exp\left(\binom{n}{\frac{n}{2}-1}\left(\frac{1}{2^{\frac{n}{2}}} + \frac{n^2}{2^{n+5}} - \frac{n}{2^{n+4}}\right)\right)$$

for even n and

$$\psi(n) \sim 2 \cdot 2^{\binom{n}{(n-1)/2}} \exp\left(\binom{n}{(n-3)/2} a(n) + \binom{n}{(n-1)/2} b(n)\right)$$

for odd n, where

$$a(n) = \frac{1}{2^{(n-3)/2}} - \frac{n^2}{2^{n+6}} - \frac{n}{2^{n+3}}$$

whereas

$$b(n) = \frac{1}{2^{(n+1)/2}} + \frac{n^2}{2^{n+4}}.$$

The exact formula of Dedekind numbers was obtained by Kisielewicz [17]:

$$\psi(n) = \sum_{k=1}^{2^{2^n}} \prod_{j=1}^{2^n-1} \prod_{i=0}^{j-1} \left(1 - b_i^k b_j^k \prod_{m=0}^{\log_2 i} \left(1 - b_m^i + b_m^i b_m^j\right)\right),$$

where $b_i^k = [k/2^i] - 2[k/2^{i+1}]$. Unfortunetely, it requires too much calculation to prove usable for $n > 5$.

Similar result was presented in [18]:

$$\psi(n) = \sum_{k=1}^{2^{2^n}} \prod_{j=1}^{2^n-1} \prod_{i=0}^{j-1} \left(1 - b_i^k \left(1 - b_j^k\right) \prod_{m=0}^{\log_2 i} \left(1 - b_m^i \left(1 - b_m^j\right)\right)\right).$$

Despite the differences ($(1 - b_j^k)$ instead of b_j^k) both formulas give the same values. The first one was found by counting antichains and the second by monotonic boolean functions (and this was the source of the difference).

Values of Dedekind numbers known today are presented in Table 2.

One of general methods of counting monotonic boolean functions of n variables is to divide them into smaller, disjoint groups and to count objects in every group. Two sample classification criteria [21] are presented below.

Tab. 2. Known values of Dedekind numbers.

n	$\psi(n)$	Who
0	2	R.Dedekind, 1897 [7]
1	3	R.Dedekind, 1897 [7]
2	6	R.Dedekind, 1897 [7]
3	20	R.Dedekind, 1897 [7]
4	168	R.Dedekind, 1897 [7]
5	7581	R. Church, 1940 [8]
6	7828354	M. Ward, 1946 [9]
7	2414682040998	R. Church, 1965 [19]
8	56130437228687557907788	D. Wiedemann, 1991 [20]

Tab. 3. Number of monotonic boolean functions mapping determined number of input states into 1 for $n = 3$.

k	number of functions
0	1
1	1
2	3
3	3
4	4
5	3
6	3
7	1
8	1
Total:	20

7.1. Number of input states which are mapped into 1.

Having determined the number of domain's elements (denoted by k, where $k \in \{0, \ldots, 2^n\}$) we count monotonic functions which map into 1 exactly that number of input states.

Sample values for $n = 3, 4, 5$ are presented in Tabs resp. 3, 4, 5.

As can be seen, the partition is symmetrical.

7.2. Additional parameter

Having the second parameter (determining the number of sets in the antichain) fixed, it is possible to obtain the exact formula for Dedekind number:

Tab. 4. Number of monotonic boolean functions mapping determined number of input states into 1 for $n = 4$.

k	number of functions
0	1
1	1
2	4
3	6
4	10
5	13
6	18
7	19
8	24
9	19
10	18
11	13
12	10
13	6
14	4
15	1
16	1
Total:	168

Tab. 5. Number of monotonic boolean functions mapping determined number of input states into 1 for $n = 5$.

k	number of events with k states	k	number of events with k states
0	1	17	605
1	1	18	580
2	5	19	530
3	10	20	470
4	20	21	387
5	35	22	310
6	61	23	215
7	95	24	155
8	155	25	95
9	215	26	61
10	310	27	35
11	387	28	20
12	470	29	10
13	530	30	5
14	580	31	1
15	605	32	1
16	621	Total:	7581

$$\psi(n,0) = 1,$$

$$\psi(n,1) = 2^n,$$

$$\psi(n,2) = 2^n \cdot \frac{2^n - 1}{2} - 3^n + 2^n,$$

$$\psi(n,3) = 2^n \cdot \frac{(2^n - 1)(2^n - 2)}{6} - 6^n + 5^n + 4^n - 3^n.$$

The general procedure of obtaining the formula for $\psi(n,k)$ with k fixed was presented by Kilibarda i Jovoviæ in [22] (2003). They showed formulas for $n \leqslant 10$. The problem was reduced to the issue of counting bipartite graphs with fixed number of vertices and edges and number of 2-colouring of determined type. The generalization of this method can be found in [23].

The difficulty is that the number of sets forming the antichain – on the basis of Sperner's theorem cited below – can be very large.

Definition 17 (Sperner family). *Sperner family of subsets of set X is an antichain in the partially ordered set $(P(X), \subseteq)$ (where $P(X)$ denotes the family of all subsets of X).*

Theorem 18. *If A is a Sperner family in set X then*

$$|A| \leqslant \binom{|X|}{\lfloor |X|/2 \rfloor}.$$

8. Connection between Dedekind problem and number of clusterings problem

We will show how Dedekind problem and the number of clusterings problem can be connected. In order to do that, we will express Dedekind number by means of $\vartheta(K,J)$.

Let $\mathcal{A}(n)$ denote the set of all antichains of the power set of n-element set ($\psi(n) = |\mathcal{A}(n)|$). We will divide the antichains with regard to some of their properties, namely – form of their union, existence of nonempty intersection between their elements and including singletons.

8.1. Types of antichains

We begin by identifing the following groups of antichains:

1. with elements having nonempty intersections, i.e.

$$\mathcal{A}_1(n) = \{A \in \mathcal{A}(n): \quad \exists_{a,b \in A} \quad a \cap b \neq \emptyset\}.$$

   We will denote its number by $\psi_1(n)$. Among them we will distinguish antichains:

   - not containing singletons (=: $\psi_{11}(n)$)
   - containing singletons (=: $\psi_{12}(n)$)

2. with disjoint elements (=: $\psi_2(n)$).

The total number of antichains will take form

$$\psi(n) = \psi_1(n) + \psi_2(n) = \psi_{11}(n) + \psi_{12}(n) + \psi_2(n).$$

8.2. Number of antichains of each type

8.2.1. $\psi_{11}(n)$

$\psi_{11}(n)$ determines the number of antichains in which at least two elements have nonempty intersection and no element is a singleton. Such families satisfy the proper-clusterings conditions with various K and J.

When K is determined J can not be greater than $\binom{K}{\lfloor K/2 \rfloor}$ (from Sperner theorem (18)). Let N denote this number.

The number of such families can be expressed as

$$\begin{aligned} \psi_{11}(n) &= \binom{n}{3}\vartheta(3,2) + \cdots + \binom{n}{i}\sum_{j=2}^{N}\vartheta(i,j) + \cdots + \binom{n}{n}\sum_{j=2}^{N}\vartheta(n,j) = \\ &= \sum_{i=3}^{n}\binom{n}{i}\sum_{j=2}^{N}\vartheta(i,j) \end{aligned} \tag{14}$$

Explanation:

- $\binom{n}{i}$ – choice of i elements being union of antichain elements,
- $\sum_{j=2}^{N}\vartheta(i,j)$ – possible numbers of antichain elements.

The minimimum number of elements in set for which there exists at least one antichain fulfiling required conditions is 3 hence the initial value of i is 3.

8.2.2. $\psi_{12}(n)$

Element belonging to singleton – from antichain definition – can not belong to its any other elements. Because of that fact the only thing to do to determine $\psi_{12}(n)$ with i singletons ($i = 1 \dots n$) is to multiply the number of possible choices of i singletons by $\psi_{12}(n-i)$ (the number of antichains with no singletons for properly reduced set):

$$\begin{aligned}\psi_{12}(n) &= \binom{n}{1}\psi_{11}(n-1) + \cdots + \binom{n}{i}\psi_{11}(n-i) + \cdots + \binom{n}{n}\psi_{11}(0) = \\ &= \sum_{i=1}^{n}\binom{n}{i}\psi_{11}(n-i). \end{aligned} \tag{15}$$

8.2.3. $\psi_2(n)$

The last group of antichains is formed by families which elements have no intersections. These can be counted easily using the formula for the number of partitions of k-element set (B_k denotes Bell number):

$$\begin{aligned}\psi_2(n) &= \binom{n}{1}B_1 + \cdots + \binom{n}{i}B_i + \cdots + \binom{n}{n}B_n = \\ &= \sum_{i=1}^{n}\binom{n}{i}B_i. \end{aligned} \tag{16}$$

In above sum i denotes the number of elements covered by antichain.

8.3. Final formula

Joining above formulas together we obtain:

$$\begin{aligned}\psi(n) &= \psi_{11}(n) + \psi_{12}(n) + \psi_2(n) = \\ &= \sum_{i=3}^{n}\binom{n}{i}\sum_{j=2}^{N}\vartheta(i,j) + \sum_{i=1}^{n}\binom{n}{i}\psi_{11}(n-i) + \sum_{i=1}^{n}\binom{n}{i}B_i = \\ &= \sum_{i=3}^{n}\binom{n}{i}\sum_{j=2}^{N}\vartheta(i,j) + \sum_{i=1}^{n}\binom{n}{i}\left[\sum_{k=3}^{n-i}\binom{n-i}{k}\sum_{j=2}^{N}\vartheta(k,j) + B_i\right], \end{aligned} \tag{17}$$

where B_i – is i-th Bell number.

9. Final remarks

In this paper we presented a method for counting all proper clusterings for a hierarchical classifier model with 3 clusters. This method, described in section 5., can serve as the basis for the computer assisted method for deriving the formula for (K, J) for any $J \geq 3$. Properties (P1)–(P4) give a set of properties like A–E in Lemma 11. These properties allow to derive equations, like (9)–(13), which can be used to generate conditions like (BC),(ABC),...,(ABCDE) in the case $J = 3$. For each such case a machine can check if there are any "symmetries" and what is the number of possibilities of adding a new, x_{K+1} element. A formula counting the number of proper vectors can be attached to each case. From these equations and from Lemma 7 we can easily obtain a general formula for $\vartheta(K, J)$. We also show that the considered problem is equivalent with Dedekind problem of counting Boolean functions by expressing the number of Boolean functions in terms of θ function values.

10. References

[1] Schapire R. E.; *The strength of weak learnability*, Machine Learning, 5, 1990, pp. 197–227.

[2] Eibl G., Pfeiffer K.-P.; *Multiclass boosting for weak classifiers*, Journal of Machine Learning, 6, 2005, pp. 189–210.

[3] Podolak I. T.; *Hierarchical Classifier with Overlapping class groups*, Expert Systems with Applications, 34(1), 2008, pp. 673–682.

[4] Podolak I. T.; *Hierarchical rules for a hierarchical classifier*, Adaptive and Natural Computing Algorithms, 4431, 2007, pp. 749–757.

[5] *On-Line Encyclopedia of Integer Sequences*. Available via `http://www.research.att.com/~njas/sequences`.

[6] Lipski W.; *Kombinatoryka dla programistow*, PWN, 2007.

[7] Dedekind R.; *Über Zerlegungen von Zahlen durch ihre grössten gemeinsamen Teiler*, Festschrift Hoch. Braunschweig u. ges. Werke(II), 1897, pp. 103–148.

[8] Church R.; *Numerical analysis of certain free distributive structures*, Duke Math. J., 6(3), 1940, pp. 732–734.

[9] Ward M.; *Note on the order of free distributive lattices*, Bull. Amer. Math. Soc., 52, 1946, pp. 423.

[10] Yamamoto K. A.; *A note on the order of free distributive lattices*, The Science Reports of the Kanazawa University, 2, 1953, pp. 5–6.

[11] Gilbert E. N.; *Lattice theoretic properties of frontal switching functions*, J. Math. Phys., 33(1), 1954, pp. 57–67.

[12] Yamamoto K. A.; *Logaritmic order of free distributive lattice*, J. Math. Soc. Japan, 6(3–4), 1954, pp. 343–353.

[13] Korobkov B. K.; *On monotone functions in Boolean algebra (in Russian)*, Problemy Kibernet., 13, 1965, pp. 5–28.

[14] Hansel G.; *Sur le nombre des fonctions boolennes monotones de n variables*, C.C. Acad. Sci Paris, 262(20), 1966, pp. 1088–1090.

[15] Kleitman D.; *On Dedekind's problem: The number of monotone Boolean functions*, Proc. Amer. Math. Soc., 21, 1969, pp. 677–682.

[16] Korshunov A. D.; *The number of monotone Boolean functions*, Problemy Kibernet., 38, 1981, pp. 5–108.

[17] Kisielewicz A.; *A solution of Dedekind's problem on the number of isotone Boolean functions*, Journal für die Reine und Angewandte Mathematik, 386, 1988, pp. 139–144.

[18] Tombak M., Isotam A., Tamme T.; *On Logical Method for Counting Dedekind Numbers*, Lecture Notes in Computer Science, 2138, 2001, pp. 424–427.

[19] Church R.; *Enumeration by rank of the elements of the free distributive lattice with seven generators*, Not. Amer. Math. So., 12, 1965, pp. 724.

[20] Wiedemann D.; *A computation of the eighth Dedekind number*, Order, 8, 1991, pp. 5–6.

[21] *Dedekind's Problem.* Available via `http://www.mathpages.com/home/kmath030.htm`.

[22] Kilibarda G., Jovović V.; *On the number of monotone Boolean functions with fixes number of lower units (in Russian)*, Intellektualnye sistemy, 7, 2003, pp. 193–217.

[23] Kilibarda G., Jovović V.; *Antichains of Multisets*, Journal of Integer Sequences, 7, 2007.

Received September 30, 2010