Constrained Clustering: Advances in Algorithms, Theory and Applications,
S. Basu, I. Davidson, and K. Wagstaff (Eds), Chapman & Hall/CRC Press,
Data Mining and Knowledge Discovery Series, pp. 145-170, 2008.

Chapter 1

Constraint-driven co-clustering of 0/1 data

Ruggero G. Pensa
KDDLab, ISTI-CNR, Area della Ricerca di Pisa, I-56124 Pisa, Italy
ruggero.pensa@isti.cnr.it

Céline Robardet
INSA-Lyon, LIRIS CNRS UMR5205, F-69621 Villeurbanne cedex, France
celine.robardet@insa-lyon.fr

Jean-François Boulicaut
INSA-Lyon, LIRIS CNRS UMR5205, F-69621 Villeurbanne cedex, France
jean-francois.boulicaut@insa-lyon.fr

Abstract We investigate a co-clustering framework (i.e., a method that provides a partition of objects and a linked partition of features) for binary data sets. So far, constrained co-clustering has been seldomly explored. First, we consider straightforward extensions of the classical instance level constraints (Must-link, Cannot-link) to express relationships on both objects and features. Furthermore, we study constraints that exploit sequential orders on objects and/or features. The idea is that we can specify whether the extracted co-clusters should involve or not contiguous elements (Interval and non-Interval constraints). Instead of designing constraint processing integration within a co-clustering scheme, we propose a Local-to-Global (L2G) framework. It consists in postprocessing a collection of (constrained) local patterns that have been computed beforehand (e.g., closed feature sets and their supporting sets of objects) to build a global pattern like a co-clustering. Roughly speaking, the algorithmic scheme is a K-Means-like approach that groups the local patterns. We show that it is possible to push local counterparts of the global constraints on the co-clusters during the local pattern mining phase itself. A large part of the chapter is dedicated to experiments that demonstrate the added-value of our approach. Considering both synthetic data and real gene expression data sets, we discuss the use of constraints to get not only more stable but also more relevant co-clusters.

0-8493-0052-5/00/$0.00+$.50

1.1 Introduction

Many data mining techniques have been proposed to support knowledge discovery from large 0/1 data sets, i.e., Boolean matrices whose rows denote objects and columns denote Boolean attributes recording object properties. Tab. 1.1(a) gives an example of such a matrix where, for instance, object t_2 only satisfies properties g_2 and g_5. In fact, many application domains provide such data sets: beside data sets that are intrinsically Boolean, they can be applied on categorical or numerical data sets as well. Indeed, a categorical feature that can have n different values can be transformed into n Boolean attributes, and continuous attributes can also be transformed into a set of Boolean attributes using discretization methods [14]. Just to name a few interesting application domains, objects can denote commercial transactions, WWW sessions, digital documents, or biological samples. Properties could then denote purchased items (i.e., a product belongs to the transaction or not), WWW resources (i.e., a resource has been uploaded or not during a session), keywords (i.e., a keyword is considered as a descriptor or not for the content of a document), or gene expression properties (e.g., a given gene has been found over-expressed or not in a biological sample).

Exploratory data analysis processes often make use of clustering techniques. This can be used to look for groups of similar objects according to some metrics. Properties can be considered as well. Many methods can provide relevant partitions on one dimension (say objects or properties) but they suffer from the lack of explicit cluster characterization, i.e., what are the properties that are shared by the objects of a same cluster. This has motivated the research on conceptual clustering and, among others, the design of co-clustering algorithms [24, 13, 9, 3]. The goal of a co-clustering task is to cluster simultaneously the rows and columns of the data matrix such that there is a one to one mapping that associates to a cluster of objects the cluster of properties that are mostly supported by these objects. An example of a co-clustering result on the data of Tab. 1.1(a) would be $\{\{\{t_1, t_3, t_4\}, \{g_1, g_3, g_4\}\}, \{\{t_2, t_5, t_6, t_7\}, \{g_2, g_5\}\}\}$. The first co-cluster indicates that the objects from cluster $\{t_1, t_3, t_4\}$ almost always share properties from cluster $\{g_1, g_3, g_4\}$. Also, properties in $\{g_2, g_5\}$ are characteristic of objects in $\{t_2, t_5, t_6, t_7\}$ (see Tab. 1.1(b)).

In the context of, for instance, WWW usage mining, this kind of pattern may help to identify communities, i.e., groups of users whose sessions give rise to almost the same uploading behavior. This is also useful in the context of gene expression data analysis where such co-clusters may provide putative synexpression groups of genes [5].

As clustering algorithms, co-clustering methods heuristically optimize an objective function (e.g., Goodman-Kruskal's τ coefficient [24] or the loss of mutual information [13]), the search space being too large to enable exhaustive

TABLE 1.1: A Boolean matrix **r** (a) and its associated co-clustering (b).

	g_1	g_2	g_3	g_4	g_5
t_1	1	0	1	1	0
t_2	0	1	0	0	1
t_3	1	0	1	1	0
t_4	0	0	1	1	0
t_5	1	1	0	0	1
t_6	0	1	0	0	1
t_7	0	0	0	0	1

(a)

	g_1	g_3	g_4	g_2	g_5
t_1	1	1	1	0	0
t_3	1	1	1	0	0
t_4	0	1	1	0	0
t_2	0	0	0	1	1
t_5	1	0	0	1	1
t_6	0	0	0	1	1
t_7	0	0	0	0	1

(b)

computation. In such a context, it is mandatory to take advantage of any available information to guide the search. Thus, a timely challenge is to support constrained co-clustering, where user-defined constraints can be used to reduce the search space. .

Our contribution is twofold. First, we are not aware of previous work related to constrained co-clustering. Not only we extend the use of *Must-link* and *Cannot-link* constraints [27, 16, 4, 11, 10, 26] within a co-clustering task, but also we introduce new constraints that are useful when the object and/or property dimensions are ordered (e.g., when properties are measured at several time steps giving rise to objects that are ordered w.r.t. time). Thanks to our *Interval* and *Non-interval* constraints, it is possible to specify whether a collection of co-clusters has to be consistent w.r.t. such an a priori order. In this chapter, we will briefly describe an application of such constraints for gene expression data analysis. Our second contribution concerns the framework for computing the co-clusters. We recently proposed a generic method to compute co-clusters based on collections of local patterns which capture locally strong associations [21]. Such local patterns are obtained thanks to exhaustive search algorithms. Then, co-clustering is performed as an heuristic combination of these patterns using a k-means approach. We have shown that using this 2-phases process leads to more robust co-clusters. We now exploit this idea within a constrained co-clustering setting, and we show that the local pattern identification step can guide the computation of constrained co-clusters. This chapter extends the preliminary results from [22]. Not only we provide more technical details on constraint processing but also the experimental validation has been considerably extended.

The rest of the paper is organized as follows. Section 1.2 provides the problem setting, including the definition of the considered constraints. Section 1.3 recalls the framework from [21] and it introduces its extension towards constrained co-clustering. Section 1.4 concerns our experimental validation, including applications on real gene expression data sets. Section 1.5 concludes.

1.2 Problem setting

The Boolean context to be mined is $\mathbf{r} \subseteq \mathcal{T} \times \mathcal{G}$, where $\mathcal{T} = \{t_1, \ldots, t_m\}$ is a set of objects and $\mathcal{G} = \{g_1, \ldots, g_n\}$ is a set of Boolean properties. We assume that $r_{ij} = 1$ if property g_j is satisfied by object t_i. For the sake of clarity, $\mathcal{D}$ will denote either $\mathcal{T}$ or $\mathcal{G}$. Let us now define the co-clustering task.

DEFINITION 1.1 Co-clustering task *A co-clustering task delivers a partition $\Pi_{\mathcal{T}}$ of k clusters of objects $\{\pi_{\mathcal{T}1}, \ldots, \pi_{\mathcal{T}k}\}$ and a partition $\Pi_{\mathcal{G}}$ of k clusters of properties $\{\pi_{\mathcal{G}1}, \ldots, \pi_{\mathcal{G}k}\}$ with a bijective mapping denoted σ between both partitions:*

$$\sigma : \Pi_{\mathcal{T}} \rightarrow \Pi_{\mathcal{G}}$$

The computed co-clustering, denoted Π, is composed of k co-clusters $\{\pi_1, \ldots, \pi_k\}$ with $\pi_i = (\pi_{\mathcal{T}i}, \sigma(\pi_{\mathcal{T}i}))$.

Example 1.1
An example of co-clustering is presented in Tab. 1.1(b). It is composed of the two partitions:

$$\Pi_{\mathcal{T}} = \{\{t_1, t_3, t_4\}, \{t_2, t_5, t_6, t_7\}\}$$
$$\Pi_{\mathcal{G}} = \{\{g_1, g_3, g_4\}, \{g_2, g_5\}\}$$

The associated function σ is defined as:

$$\sigma(\{t_1, t_3, t_4\}) = \{g_1, g_3, g_4\}$$
$$\sigma(\{t_2, t_5, t_6, t_7\}) = \{g_2, g_5\}$$

□

In such a framework, it makes sense to apply the standard Must-link and Cannot-link constraints on both object and property sets.

DEFINITION 1.2 Extended Must-link and Cannot-link constraints *An extended Must-link constraint, denoted $c_{e=}(x_i, x_j, \Pi, \mathcal{D})$, specifies that two elements x_i and x_j of $\mathcal{D}$ have to belong to a same co-cluster from Π. An extended Cannot-link constraint, denoted $c_{e\neq}(x_i, x_j, \Pi, \mathcal{D})$, specifies that x_i and x_j can not belong to the same co-cluster of Π.*

Assume now there exists a function $s : \mathcal{D} \rightarrow \mathbb{R}$ that associates a real value $s(x_i)$ to each element $x_i \in \mathcal{D}$. For instance, $s(x_i)$ can be a temporal or spatial measure related to x_i. For instance, in microarray data, where $\mathcal{T}$ is a set of DNA chips corresponding to biological experiments, and $\mathcal{G}$ is a set of genes,

$s(t_i)$ might be the time stamp for the experiment t_i. Such a function s enables to define an order $\preceq$ on dimension $\mathcal{D}$ where $x_i \preceq x_j$ iff $s(x_i) \leq s(x_j)$. For the sake of simplicity, we consider that if a function s exists on dimension $\mathcal{D}$, then all its elements x_i are ordered. We can now introduce constraints that exploit an ordered set $\mathcal{D}$.

DEFINITION 1.3 Interval and Non-interval constraints *Given an order ($\preceq$) on $\mathcal{D}$, an Interval constraint on this dimension, denoted $c_{int}(\mathcal{D}, \Pi)$, enforces each cluster on $\mathcal{D}$ to be an interval: $\forall \ell = 1 \dots k$, if $x_i, x_j \in \pi_{\mathcal{D}\ell}$ then $\forall x$ s.t. $x_i \preceq x \preceq x_j$, $x \in \pi_{\mathcal{D}\ell}$. A Non-interval constraint denoted $c_{non-int}(\mathcal{D}, \Pi)$ specifies that clusters on $\mathcal{D}$ should not be intervals: $\forall \ell = 1 \dots k$, $\exists x_i, x_j \in \pi_{\mathcal{D}\ell}$, $\exists x \in \mathcal{D}$ s.t. $x_i \preceq x \preceq x_j$, $x \notin \pi_{\mathcal{D}\ell}$.*

An Interval constraints can be used to find clusters which are continuous intervals, while a Non-interval constraints can be used to find clusters which are not intervals.

Example 1.2
Suppose that $\mathcal{G}$ is ordered such that $g_1 \preceq g_2 \preceq g_3 \preceq g_4 \preceq g_5$. Partition $\Pi_{\mathcal{G}} = \{\{g_1, g_3, g_4\}, \{g_2, g_5\}\}$ does not satisfy $c_{int}(\mathcal{D}, \Pi_{\mathcal{G}})$ since g_2 and g_5 belongs to $\pi_{\mathcal{G}2}$ but g_3 ($g_2 \preceq g_3 \preceq g_5$) does not belong to $\pi_{\mathcal{G}2}$.
$c_{non-int}(\mathcal{D}, \Pi_{\mathcal{G}})$ is an example of a satisfied constraint. □

One of the typical application domains which motivates the use of these constraints is temporal gene expression data analysis: objects are a given organism considered at several developmental steps and properties encode, for instance, the over-expression of genes. In such a context (see, e.g., [2, 8]), using Interval constraints enables to capture conjunctions of properties which characterize any single developmental period, while the use of Non-interval constraints might point out interactions which are somehow time-independent.

1.3 A constrained co-clustering algorithms based on a Local-to-Global approach

In [21], we have proposed a generic co-clustering framework. The main idea is to compute a co-clustering not starting from the raw data but from local patterns that capture locally strong associations between sets of objects and sets of properties. Let us first consider the Local-to-Global (L2G) aspect of our proposal. We use the simple formalization introduced in [12] to support the discussion.

1.3.1 A Local-to-Global approach

Many local pattern mining techniques (e.g., looking for frequent patterns, data dependencies) have been studied extensively the last decade (see, e.g., [20] for a recent survey on local pattern detection issues). Many mining tasks can be formalized as the computation of the theory $Th(\mathcal{L}, \mathbf{r}, q) = \{\phi \in \mathcal{L} \mid q(\phi, \mathbf{r}) \text{ is true}\}$ where $\mathbf{r}$ is the data, $\mathcal{L}$ is a language of patterns, and $q(\phi, \mathbf{r})$ denotes the selection predicate of interesting patterns [19]. One crucial observation with this simple formalization is that the interestingness of a pattern can be tested independently of the other patterns (e.g., testing that a pattern is frequent enough can be done without looking at other solution patterns). For us, in such a context, it justifies that instances of this framework are called local pattern mining tasks.

However, many useful mining tasks are looking for models or global patterns (e.g., classifiers, clusterings) and can be formalized as the computation of sets of patterns that satisfy constraints. Let us assume that lower case letters such as ϕ denote individual patterns and that upper case ones as Φ denote sets of patterns, a Local-to-Global mining task can be defined by the two following steps:

1. $L = Th(\mathcal{L}, \mathbf{r}, q)$
2. $M = Th(f(L), \mathbf{r}, p)$ where $f(L)$ is a transformation of L,

$$Th(f(L), \mathbf{r}, p) = \{\Phi \subseteq f(L) \mid p(\Phi, \mathbf{r}) \text{ is true}\}$$

In this context, the constraint q is said to be local as it applies on individual pattern, whereas the constraint p is said global as it has to hold for a set of patterns. The popular association-based classification approach [17] is an obvious example of a L2G scheme: standard association rules are the local patterns (i.e., local constraints are the minimal frequency and minimal confidence constraints). The various proposals for building classifiers from them are then based on different global constraints on these collections of association rules. Clustering can be considered within a L2G framework as well.

1.3.2 The CDK-Means proposal

The CDK-Means algorithm is our L2G proposal for co-clustering 0/1 data. It is closely related to subspace clustering and it exploits the many results that are available for local pattern extraction from large Boolean matrices.

Our language of patterns is the language of bi-sets and it is denoted $\mathcal{B}$:

$$\mathcal{B} \equiv \{(T, G) \mid T \subseteq \mathcal{T} \text{ and } G \subseteq \mathcal{G}\}$$

A bi-set $b = (T, G)$ is thus a couple made of a set of objects and a set of properties. Clearly, many interesting classes of bi-sets can be computed from a

data set $\mathbf{r}$ given a selection predicate q. For instance, the selection predicate can enforce that the set of properties is a frequent itemset because the set of objects that share these properties has a size greater than a user-defined threshold [1]. We can use a more selective predicate and, for instance, restrict the bi-sets to the ones whose sets of properties are closed sets. In that case, we are looking for the well known formal concepts (see, e.g., [7]). Many other interesting types of bi-sets could be considered, e.g., support envelopes [25] or the dense and relevant bi-sets [6]. Discussing the pros and cons of each type of local pattern is however out of the scope of such a chapter. Let us notice that we find in the literature (see the survey in [18]) various local pattern mining tasks that are named bi-clustering tasks as soon as both dimensions (objects and properties) are involved. In our case, we prefer to talk about a clustering only in the context of unsupervised classification and thus the computation of collections of clusters, not collections of local patterns whose interestingness can be evaluated based on individuals only.

The CDK-Means algorithm introduced in [21] computes

$$Th(\mathcal{B}, \mathbf{r}, p_{CDK}) = \{\Phi \subseteq \mathcal{B} \mid p_{CDK}(\Phi, \mathbf{r}) \text{ is true}\}$$

with Φ being a partition Π_B of a bi-set collection B ($\Pi_B : B \rightarrow \{1 \cdots k\}$) and

$$p_{CDK}(\Pi_B, \mathbf{r}) \quad \equiv \quad \Pi_B = \text{argmin} \sum_{b_j \in B} d(b_j, \mu_{\Pi_B(b_j)})$$

The constraint p_{CDK} is the one used in the K-Means algorithm with d being a distance and $\mu_{\Pi_B(b_j)}$ being the centroid of the cluster that contains the bi-set b_j. Let us now introduce some notations to formally define these quantities.

First, we describe each bi-set b_j by its characteristic vector as follows:

$$< \mathbf{t}_j >, < \mathbf{g}_j > = < t_{j1}, \ldots, t_{jm} >, < g_{j1}, \ldots, g_{jn} >$$

where $t_{ji} = 1$ if $t_i \in T_j$ (0 otherwise) and $g_{ji} = 1$ if $g_i \in G_j$ (0 otherwise).

We are looking for k clusters of bi-sets $\{\pi_{B1}, \ldots, \pi_{Bk}\}$ ($\pi_{B\ell} \subseteq B$). Let us define the centroid of a cluster of bi-sets $\pi_{B\ell}$ as $\mu_\ell = < \tau_\ell >, < \gamma_\ell > = < \tau_{\ell 1}, \ldots, \tau_{\ell m} >, < \gamma_{\ell 1}, \ldots, \gamma_{\ell n} >$ where τ and γ are the usual centroid components:

$$\tau_{\ell i} = \frac{1}{|\pi_{B\ell}|} \sum_{b_j \in \pi_{B\ell}} t_{ji}, \quad \gamma_{\ell i} = \frac{1}{|\pi_{B\ell}|} \sum_{b_j \in \pi_{B\ell}} g_{ji}$$

We now define the distance used between a bi-set b_j and a centroid μ_ℓ:

$$d(b_j, \mu_\ell) = \frac{1}{2} \left(\frac{|\mathbf{t}_j \cup \boldsymbol{\tau}_\ell| - |\mathbf{t}_j \cap \boldsymbol{\tau}_\ell|}{|\mathbf{t}_j \cup \boldsymbol{\tau}_\ell|} + \frac{|\mathbf{g}_j \cup \boldsymbol{\gamma}_\ell| - |\mathbf{g}_j \cap \boldsymbol{\gamma}_\ell|}{|\mathbf{g}_j \cup \boldsymbol{\gamma}_\ell|} \right)$$

It is the mean of the weighted symmetrical differences of the set components. We assume $|\mathbf{t}_j \cap \boldsymbol{\tau}_\ell| = \sum_{i=1}^{m} a_i \frac{t_{ji} + \tau_{\ell i}}{2}$ and $|\mathbf{t}_j \cup \boldsymbol{\tau}_\ell| = \sum_{i=1}^{m} \frac{t_{ji} + \tau_{\ell i}}{2}$ where $a_i = 1$ if $t_{ji} \cdot \tau_{\ell i} \neq 0$, 0 otherwise. Intuitively, the intersection is equal to the

TABLE 1.2: CDK-MEANS pseudo-code.

CDK-MEANS (**r** is a Boolean context, B is a collection of bi-sets in **r**, k is the number of clusters, MI is the maximal iteration number.)

1. Let $\mu_1 \dots \mu_k$ be the initial cluster centroids. $it := 0$.
2. Repeat
   (a) For each bi-set $b_j \in B$, assign it to cluster $\pi_{B\ell}$ s.t. $d(b_j, \mu_\ell)$ is minimal.
   (b) For each cluster $\pi_{B\ell}$, compute $\boldsymbol{\tau_\ell}$ and $\boldsymbol{\gamma_\ell}$.
   (c) $it := it + 1$.
3. Until centroids are unchanged or $it = MI$.
4. For each $t_i \in \mathcal{T}$ (resp. $g_i \in \mathcal{G}$), assign it to the first cluster $\pi_{\mathcal{T}\ell}$ (resp. $\pi_{\mathcal{G}\ell}$) s.t. $\tau_{\ell i} = \frac{1}{|\pi_{B\ell}|} \sum_{b_j \in \pi_{B\ell}} t_{ji}$ (resp. $\gamma_{\ell i} = \frac{1}{|\pi_{B\ell}|} \sum_{b_j \in \pi_{B\ell}} g_{ji}$) is maximum.
5. Return $\{\pi_{\mathcal{T}1} \dots \pi_{\mathcal{T}k}\}$ and $\{\pi_{\mathcal{G}1} \dots \pi_{\mathcal{G}k}\}$

mean between the number of common objects and the sum of their centroid weights. The union is the mean between the number of objects and the sum of their centroid weights. These measures are defined similarly on properties.

Objects t_i (resp. properties g_i) are assigned to one of the k clusters (say cluster ℓ) for which $\tau_{\ell i}$ (resp. $\gamma_{\ell i}$) is maximum. We can enable that a number of objects and/or properties belong to more than one cluster by controlling the size of the overlapping part of each cluster. Thanks to our definition of cluster membership determined by the values of $\boldsymbol{\tau_\ell}$ and $\boldsymbol{\gamma_\ell}$, we just need to adapt the cluster assignment step given some user-defined thresholds.

A simplified algorithm CDK-MEANS is given in Tab. 1.2: for the sake of brevity, we do not consider further cluster overlapping. It computes a co-clustering of **r** given a collection of bi-sets B extracted from **r** beforehand (e.g., collections of formal concepts). CDK-MEANS can provide the example co-clustering given in Section 1.

1.3.3 Constraint-driven co-clustering

Let us now consider the L2G framework when some of the co-clustering constraints defined in Section 1.2 have been specified (i.e., beside the specification of the number of co-clusters and the implicit optimization constraint on the objective function, the global constraint might also contain Must-link, Cannot-link, Interval or Non-interval constraints).

The key idea is that, to compute a co-clustering that satisfies the specified

global constraints, we can exploit *local counterparts* of them, i.e., constraints that apply on local patterns (here, bi-sets) to select only part of them. For some constraints (e.g., Must-link), the satisfaction of the local counterpart is sufficient to guarantee that the co-clusters satisfy the global constraint. In such cases, we say that the local constraint is automatically propagated to the global level. For other constraints, it happens that a new global constraint must be used in addition to the local counterpart. In such cases however, we consider new global constraints that are easier to check. Notice also that, given the state-of-the-art, evaluating local constraints can be extremely more efficient than checking for global ones, and quite efficient algorithms are available to compute bi-sets that satisfy, e.g., monotonic constraints (see Section 1.3.4). For instance, we use here D-MINER [7] to computate collections of formal concepts that satisfy size constraints on both object and property sets. Let us now provide more details on how we manage to process the different constraints.

- The *local counterpart* of a Must-link constraint $c_{e=}(x_i, x_j, \Pi, \mathcal{D})$ consists in selecting bi-sets that contains either both x_i and x_j or none of them:

$$q_{Must-link}(b, \mathbf{r}) \quad \equiv \quad (x_i \in b \wedge x_j \in b) \vee ((x_i \notin b \wedge x_j \notin b)$$

  As the coefficients of each object/property in each centroid μ_ℓ depends on the number of bi-sets containing this object/property, the coefficients corresponding to x_i and x_j are equal in each centroid. Consequently, x_i and x_j are assigned to the same cluster and thus the local counterpart of the Must-link constraint is automatically propagated to the computed co-clusters.

- A necessary condition for the extended Cannot-link constraint, is that B does not contain any bi-set violating the constraint. Indeed, the *local counterpart* of a Cannot-link constraint $c_{e\neq}(x_i, x_j, \Pi, \mathcal{D})$ consists in selecting bi-sets that do not contain both x_i and x_j:

$$q_{Cannot-link}(b, \mathbf{r}) \quad \equiv \quad \neg(x_i \in b \wedge x_j \in b)$$

  This condition does not ensure that the global Cannot-link constraint is satisfied in the final co-clustering (it is not automatically propagated) and a further control is needed. In particular, in Step (2a) of CDK-MEANS algorithm (see Tab. 1.2), before adding a bi-set containing x_i (resp. x_j) to a cluster, we should ensure that no bi-set containing x_j (resp. x_i) has been assigned to it earlier.

- For the Interval and Non-interval constraints, it is possible to directly use these constraints on each bi-set independently, i.e., as a local constraint. However, for the Interval constraint, it might be too stringent in practice (i.e., too few bi-sets would satisfy the constraint), whereas for the Non-interval one, it will not be selective enough (almost all the

bi-sets would satisfy the constraint). For these reasons, we propose to relax the Interval constraint and to strengthen the Non-interval constraints as *local counterparts* of the global ones. These two new local constraints, respectively called Max-gap and Min-gap, are now defined.

DEFINITION 1.4 Max-gap and Min-gap constraints *Given an order on $\mathcal{D}$, a Max-gap constraint on $\mathcal{D}$, denoted $c_{maxgap}(\mathcal{D}, \sigma, b)$, is satisfied by b w.r.t threshold σ iff, for each pair of consecutive elements $x_i, x_j \in b$ s.t. $x_i \prec x_j$, $|\{x_h \notin b$, with $x_i \prec x_h \prec x_j\}| \leq \sigma$. A Min-gap constraint, denoted $c_{mingap}(\mathcal{D}, \sigma, b)$, is satisfied by b w.r.t threshold σ iff, for each pair of consecutive elements $x_i, x_j \in b$ s.t. $x_i \prec x_j$, $|\{x_h \notin b$, with $x_i \prec x_h \prec x_j\}| \geq \sigma$.*

Max-gap is used as a *local counterpart* of the Interval constraint and Min-gap as the Non-interval one. Clearly, these local constraints do not ensure the satisfaction of c_{int} and $c_{non-int}$, but it supports the computation of more relevant co-clusters (see the experimental section).

Example 1.3
Suppose that $\mathcal{G}$ is ordered such that $g_1 \preceq g_2 \preceq g_3 \preceq g_4 \preceq g_5$. The bi-set $b = \{\{t_1, t_3, t_4\}, \{g_1, g_3, g_4\}\}$ satisfies $c_{maxgap}(\mathcal{G}, 1, b)$ but not $c_{maxgap}(\mathcal{G}, 0, b)$. It does not satisfy $c_{mingap}(\mathcal{G}, 1, b)$. □

The modified version of the algorithm is sketched in Tab. 1.3 (modifications in bold). It uses the same greedy strategy than COP-KMeans [27]. A slight difference is that, as we process local patterns (bi-sets), then it concerns only the sets of objects and properties that belong to the local patterns.

1.3.4 Discussion on constraint processing

To summarize, we can classify the constraints w.r.t. propagation issues.

1. Constraints that are automatically propagated from the local level to the global one (extended Must-link).

2. Constraints that need for a control to be propagated from the local level to the global one (extended Cannot-link).

3. Constraint whose propagation to the local level to the global one is not ensured (Interval and Non-interval constraints).

Even if we cannot ensure the ultimate satisfaction for the Interval and Non-interval constraints, we show in our experimental validation (see Section 1.4) that using Max-gap and Min-gap local constraints enables to produce clusters

TABLE 1.3: Constrained CDK-MEANS pseudo-code.

CDK-MEANS (**r** is a Boolean context, B is a collection of local patterns in **r**, k is the number of clusters, MI is the maximal iteration number, C **a set of constraints**.)

1. **Let $B' \subseteq B$ be the sub-collection satisfying all the local counterparts of the constraints in C (extended Must-link and Cannot-link, Max-gap and Min-gap).**
2. Let $\mu_1 \dots \mu_k$ be the initial cluster centroids. $it := 0$.
3. Repeat
   (a) For each local pattern $b_j \in B'$, assign it to cluster $\pi_{B\ell}$ s.t. $d(b_j, \mu_\ell)$ is minimum and **no Cannot-link constraint is violated.**
   (b) For each cluster $\pi_{B\ell}$, compute $\boldsymbol{\tau_\ell}$ and $\boldsymbol{\gamma_\ell}$.
   (c) $it := it + 1$.
4. Until centroids are unchanged or $it = MI$.
5. For each $t_i \in \mathcal{T}$ (resp. $g_i \in \mathcal{G}$), assign it to the first cluster $\pi_{\mathcal{T}\ell}$ (resp. $\pi_{\mathcal{G}\ell}$) s.t. $\tau_{\ell i}$ (resp. $\gamma_{\ell i}$) is maximum.
6. Return $\{\pi_{\mathcal{T}1} \dots \pi_{\mathcal{T}k}\}$ and $\{\pi_{\mathcal{G}1} \dots \pi_{\mathcal{G}k}\}$

that tend to be intervals. One of the perspectives of this work is clearly to enforce the propagation of these two constraints by introducing a control step in the iterative part of the algorithm (as for the extended Cannot-link constraint).

When a constraint is too selective (too many bi-sets violate such a constraint), some objects and/or properties could not be represented in the collection used by our algorithm. The solution of a garbage cluster for each unclustered object/property is not always possible, specially when they are involved in Cannot-link constraints. In fact, if a Cannot-link constraint involving x_i and x_j is not satisfied by any bi-set (i.e., each time a bi-set contains x_i, it also contains x_j), the two objects/properties will not be contained in the final co-clustering. But they cannot be assigned to a garbage co-cluster, since a Cannot-link constraint prevents this situation. If such a situation happens, our approach returns no solution. We say that the co-clustering is unfeasible. Notice that in this work we do not address the problem of the feasibility of conjunctions of Must-link and Cannot-link constraints (see [11] for a complete overview of the constraint feasibility problem).

As discussed earlier, the local counterpart of a global constraint is easier to check, especially when the local constraint is monotonic w.r.t. some specialization relation. In such cases, we can use efficient local pattern mining algorithms. For bi-set mining, we use D-MINER [7] that exploit such constraints.

DEFINITION 1.5 D-MINER ***specialization and monotonicity*** *The specialization on bi-sets from $\mathcal{B}$ used by* D-MINER *is defined by $(T_1, G_1) \leq (T_2, G_2)$ iff $T_1 \subseteq T_2$ and $G_1 \subseteq G_2$. A constraint $\mathcal{C}$ is said anti-monotonic w.r.t. $\leq$ iff $\forall \alpha, \beta \in \mathcal{B}$ such that $\alpha \leq \beta$, $\mathcal{C}(\beta) \Rightarrow \mathcal{C}(\alpha)$. $\mathcal{C}$ is said monotonic w.r.t. $\leq$ iff $\forall \alpha, \beta \in \mathcal{B}$ such that $\alpha \leq \beta$, $\mathcal{C}(\alpha) \Rightarrow \mathcal{C}(\beta)$.*

D-MINER efficiently exploits both monotonic and anti-monotonic constraints. Consequently, if the local constraints used are conjunctions or disjunctions of monotonic constraints, they can be directly pushed during the local pattern mining step.

Let us now summarize how the local counterparts of the co-clustering constraints can be directly used during our bi-set mining step.

- The local counterparts of the extended Must-link and Cannot-link constraints require that some elements are included or not in bi-sets (see $q_{Must-link}(b, \mathbf{r})$ and $q_{Cannot-link}(b, \mathbf{r})$ constraints). Such constraints are monotonic while considering the specialization relation of Definition 1.5. Consequently, the local counterparts of these constraints can be represented by a conjunction and/or a disjunction of monotonic constraints that can be exploited by D-MINER.

- The Min-gap constraint is anti-monotonic. Let $b_1 = (X_1, Y_1)$ and $b_2 = (X_2, Y_2)$ be two bi-sets s.t. $X_1 \subseteq X_2$. If we define $S_i = \{x_h \notin X_i | x_i \prec x_h \prec x_j\}$, we have $S_2 \subseteq S_1$. Consequently, if $|S_2| \geq \sigma$, i.e., $c_{maxgap}(\mathcal{D}, \sigma, b_2)$ is satisfied, then $|S_1| \geq \sigma$ and $c_{maxgap}(\mathcal{D}, \sigma, b_1)$ is satisfied as well.

- The Max-gap constraint has no monotonicity properties. Let us consider the Max-gap constraint $c_{maxgap}(D, 1, b)$ with $D = \{x_1, x_2, \ldots, x_n\}$. It is not satisfied by $X_1 = \{x_2, x_3, x_7\}$ but it is satisfied by $X_2 = \{x_2, x_3, x_5, x_7\}$ and one of its subset $X_0 = \{x_2, x_3\}$. Therefore, it is neither monotonic nor anti-monotonic. This constraint is thus checked after the execution of the bi-set mining algorithm, i.e., in a post-processing phase.

1.4 Experimental validation

Let us first introduce some measures which are to be used in our experiments to evaluate the quality of co-clustering results. In the first batch of experiments, we show the interest in using pairwise constraints on both dimensions thanks to an application to a synthetic data set for which standard and unconstrained co-clustering approach produce unstable results. Then we apply our framework on two real temporal gene expression data sets to illustrate the added-value of Interval constraints. Finally, we discuss another application to another gene expression data set to illustrate how the Interval and Non-interval constraints can be used to supervise a co-clustering task to discover more stable co-clusters which are more relevant as well.

1.4.1 Evaluation method

A general criterion to evaluate clustering results consists in comparing the computed partition with a "correct" one. It means that data instances are already associated to some correct labels and that we want to quantify the agreement between computed labels and correct ones. A popular measure is the Rand index which measures the agreement between two partitions of m elements. If $\mathbf{C} = \{C_1 \dots C_s\}$ is our clustering structure and $\mathbf{P} = \{P_1 \dots P_t\}$ is a predefined partition, each pair of data points is either assigned to the same cluster in both partitions or to different ones. Let a be the number of pairs belonging to the same cluster of $\mathbf{C}$ and to the same cluster of $\mathbf{P}$. Let b be the number of pairs whose points belong to different clusters of $\mathbf{C}$ and to different clusters of $\mathbf{P}$. The agreement between $\mathbf{C}$ and $\mathbf{P}$ can be estimated using

$$Rand(\mathbf{C}, \mathbf{P}) = \frac{a+b}{m \cdot (m-1)/2}$$

It takes values between 0 and 1 and it is maximized when $s = t$ [23].

To evaluate the added-value of the Interval constraint, we propose to measure the number of jumps within a partition.

DEFINITION 1.6 Jump number *Given $\mathcal{D} = \{x_1, \dots, x_n\}$ a set of ordered points and a cluster $\pi_{\mathcal{D}\ell}$ on these points, we have a jump given a number $\nu > 1$, if $x_i \in \pi_{\mathcal{D}\ell}$, $x_{i+\nu} \in \pi_{\mathcal{D}\ell}$ and $\forall h$ s.t. $i < h < i + \nu$, $x_h \notin \pi_{\mathcal{D}\ell}$. Let J_ℓ be the number of jumps within a cluster $\pi_{\mathcal{D}\ell}$. Given a partition $\Pi_\mathcal{D} = \{\pi_{\mathcal{D}1}, \dots, \pi_{\mathcal{D}k}\}$, the jump number measure denoted N_J is then*

$$N_J = \sum_{\pi_{\mathcal{D}\ell} \in \Pi_\mathcal{D}} J_\ell$$

.

If $N_J = 0$, clusters are intervals. As the Interval constraint is processed as a soft constraint, we average the N_J measure on a set of clustering instances (with random initialization) to measure the efficiency of the approach.

We also want to evaluate co-clustering quality by means of an internal criterion. An interesting measure for this purpose is the symmetrical Goodman and Kruskal's τ coefficient [15] which evaluates the proportional reduction in error given by the knowledge of $\Pi_{\mathcal{T}}$ on the prediction of $\Pi_{\mathcal{G}}$ and vice versa. Another measure is the loss in mutual information [13], which is the objective function that COCLUSTER tries to minimize. Both coefficients are evaluated on a contingency table $\mathbf{p}$. Let p_{ij} be the frequency of relations between an object of a cluster $\pi_{\mathcal{T}i}$ and a property of a cluster $\pi_{\mathcal{G}j}$. Furthermore, we have $p_{i.} = \sum_j p_{ij}$ and $p_{.j} = \sum_i p_{ij}$. The Goodman-Kruskal's τ coefficient is defined as follows:

$$\tau = \frac{\frac{1}{2}\sum_i \sum_j \left(p_{ij} - p_{i.}p_{.j}\right)^2 \frac{p_{i.}+p_{.j}}{p_{i.}p_{.j}}}{1 - \frac{1}{2}\sum_i p_{i.}^2 - \frac{1}{2}\sum_j p_{.j}^2}$$

The mutual information, which computes the amount of information $\Pi_{\mathcal{T}}$ contains about $\Pi_{\mathcal{G}}$ is:

$$I(\Pi_{\mathcal{T}}, \Pi_{\mathcal{G}}) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{p_{i.}p_{.j}}$$

Then, given two different co-clusterings $(\Pi_{\mathcal{T}}, \Pi_{\mathcal{G}})$ and $(\hat{\Pi}_{\mathcal{T}}, \hat{\Pi}_{\mathcal{G}})$, the loss in mutual information is given by:

$$I(\Pi_{\mathcal{T}}, \Pi_{\mathcal{G}}) - I(\hat{\Pi}_{\mathcal{T}}, \hat{\Pi}_{\mathcal{G}})$$

Finally, to evaluate the performances of our method, we use a comparison coefficient which is the mean of the products between the number of needed iterations and the number of processed bi-sets, i.e.:

$$CC = \frac{\sum_i^N |B| \cdot NI_i}{N}$$

where, N is the number of executions, $|B|$ is the size of the local pattern collection computed beforehand, and NI_i is the number of iterations for the i-th execution.

1.4.2 Using extended Must-link and Cannot-link constraints

Let us use a synthetic data set which intrinsically leads to unstable clustering. The goal is here to show how using Must-link and Cannot-link constraints can support the discovery of different co-clustering structures. This synthetic data set is some kind of idealized abstraction of temporal gene expression data (see Fig. 1.1 where the gray zones denote true values in the matrix). In this data set, the expression level of 20 genes changes during the 105 time points

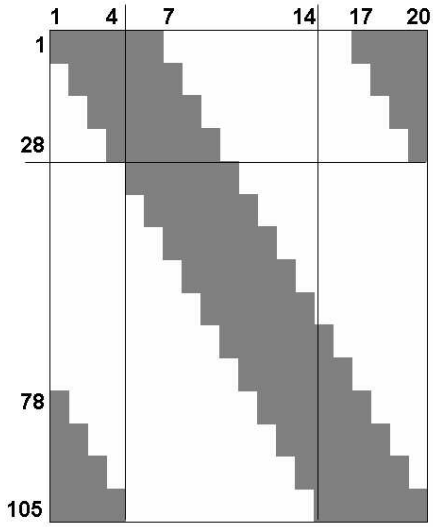


FIGURE 1.1: A synthetic data set.

of the sampling period. It is easy to capture some cyclic behavior, since the first and the last time periods gives rise to similar patterns. Such a situation should produce quite unstable clustering results.

We have extracted a collection of formal concepts using D-Miner [7]. This collection (containing 85 formal concepts) has been used for all the experiments described in this section. When using the unconstrained version of CDK-Means or Cocluster [13] (with $k = 3$), the results clearly point out two kinds of co-clustering results (see Fig. 1.2(a) and Fig. 1.2(b) for co-clusterings **a** and **b**). The second co-clustering (Fig. 1.2(b)) emphasizes some cyclic behavior, since the first and the last groups of samples are in the same co-cluster. Such a structure is clearly missing in the first co-clustering (Fig. 1.2(a)). Notice that all the results produced by the random instances of the two algorithms are similar to one of these two co-clusterings. Tab. 1.4 provide the various results concerning the τ and I values, and the Rand coefficient values computed w.r.t. the two co-clusterings for both objects and properties.

Let us assume that we want to supervise the co-clustering process to discover co-clustering **a** or co-clustering **b**. For this purpose, we can use some extended Must-link or/and Cannot-link constraints to control the search. If we look at the two targeted co-clusterings, there are objects which always belong to the same cluster, while other objects and properties "change" clusters when moving from co-clustering **a** to co-clustering **b**. For instance, objects 1 to 21 are always in the same cluster (we say they are "stable"), while objects 22 to 35 change cluster in the second co-clustering (we say they are "unstable").

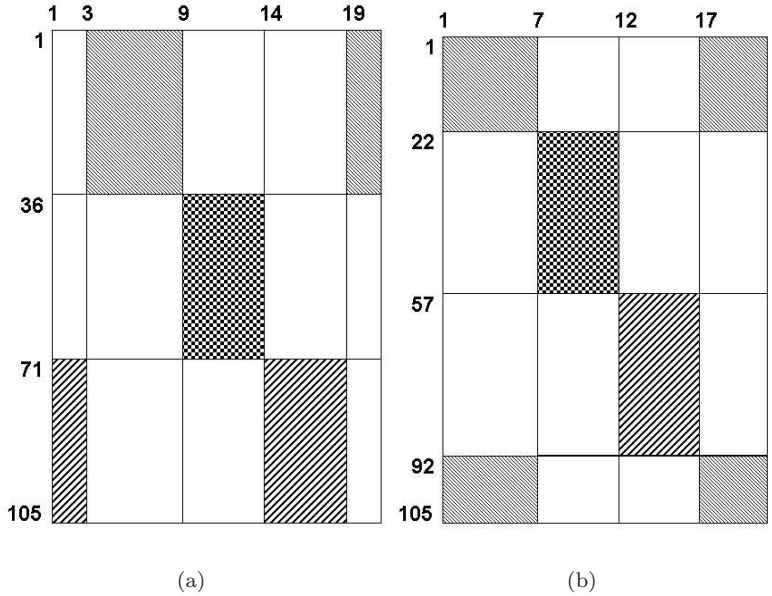


FIGURE 1.2: Two co-clusterings for the synthetic data from Fig. 1.1.

We list all possible pairs which are composed by one object belonging to the "stable" set and one object belonging to the "unstable" set. We construct a similar list for properties. Then, we pick a fixed number of pairs randomly and we compare the class labels (which are inferred from co-clusterings **a** and **a**) of each object (resp. property) inside the co-clustering we want to discover. If the objects (resp. properties) share the same class label, we construct a Must-link constraint. Otherwise, we construct a Cannot-link constraint. In our experiments, we have computed the results for 100 randomly generated sets of 1 and 2 constraints for both objects and properties. For each set of constraints, we executed 25 randomly initialized instances of CDK-Means. Results are in Tab. 1.5 (for sets of 1 constraint) and Tab. 1.6 (for sets of 2 constraints). All the average Rand indexes are 1% to 9% better than the same indexes obtained by the unconstrained versions, for both our CDK-Means algorithm or Cocluster. It works for generated constraints over both the set of objects and the set of properties. Results are also slightly more stable w.r.t. unconstrained instances. It shows that even a few number of constraints enables to obtain a co-clustering which is more stable w.r.t. initialization but also more relevant w.r.t. user expectation.

User-defined constraints can be derived from domain knowledge. Looking at the two co-clusterings (see Fig. 1.2(a) and Fig. 1.2(b), we see that the first object is clustered together with the first property in **b** but not in **a**.

TABLE 1.4: Co-clustering synthetic data without constraints (25 trials).

	Cocluster	**CDK-Means**
τ	0.29±0.03	0.34±0.02
I	0.90±0.07	0.96±0.02
a-Rand($\mathcal{T}$)	0.74±0.07	0.78±0.11
a-Rand($\mathcal{G}$)	0.74±0.08	0.77±0.11
b-Rand($\mathcal{T}$)	0.75±0.07	0.77±0.10
b-Rand($\mathcal{G}$)	0.74±0.08	0.78±0.11

Then, we can introduce an extended Cannot-link constraint between the first object and the first property to drive CDK-Means towards the first type of co-clustering. In this case, the average Rand indexes w.r.t. to **a**, computed on both objects and properties, are both equal to 0.83 (i.e., 8% better than the one obtained by unconstrained co-clustering). Now, if we add an extended Cannot-link constraint between the last object and the last property, then the two scores rise respectively to 0.87 and 0.88.

Notice however that our framework has to be considered as an unsupervised method that is useful when we lack from detailed information about the data. When a large number of constraints is introduced, some conjunctions can remove an important number of local patterns. In this case, some objects and properties might disappear from the co-clustering process.

1.4.3 Time interval cluster discovery

We have studied the impact of the Interval constraint in two microarray data sets called malaria and drosophila. The first one [8] concerns the transcriptome of the intraerythrocytic developmental cycle of Plasmodium Falciparum, i.e., a causative agent of human malaria. The data provide the expression profile of 3 719 genes in 46 biological samples. Each sample corresponds to a time point of the developmental cycle: it begins with merozoite invasion of the red blood cells, and it is divided into three main phases, the ring, trophozoite and schizont stages. The second data set is described in [2]. It concerns the gene expression of the Drosophila melanogaster during its life cycle. The expression levels of 3 944 genes are evaluated for 57 sequential time periods divided into embryonic, larval and pupal stages. The numerical gene expression data from [8] has been discretized by using one of the encoding methods described in [5]: for each gene g, we assigned the Boolean value 1 to those samples whose expression level was greater than X% of its max expression level. X was set to 25% for malaria and 35% for drosophila. The two matrices have been mined for formal concepts by using D-Miner [7].

We applied Cocluster algorithm [13] and the unconstrained version of CDK-Means with $k = 3$ to identify the three developmental stages. Since the initialization of both algorithms is randomized, we average all the measures

TABLE 1.5: Co-clustering synthetic data (1 pair-wise constraint, 100 random constraint sets, 25 trials).

	A		**B**	
	$\mathcal{T}$	$\mathcal{G}$	$\mathcal{T}$	$\mathcal{G}$
τ	0.32±0.03	0.33±0.03	0.33±0.03	0.30±0.02
I	0.97±0.04	0.96±0.03	0.97±0.03	0.96±0.03
Rand($\mathcal{T}$)	0.81±0.10	0.80±0.10	0.83±0.10	0.80±0.09
Rand($\mathcal{G}$)	0.78±0.10	0.78±0.10	0.84±0.11	0.80±0.08

TABLE 1.6: Co-clustering synthetic data (2 pair-wise constraint, 100 random constraint sets, 25 trials).

	A		**B**	
	$\mathcal{T}$	$\mathcal{G}$	$\mathcal{T}$	$\mathcal{G}$
τ	0.31±0.05	0.31±0.04	0.31±0.05	0.29±0.02
I	0.98±0.05	0.98±0.05	0.99±0.06	0.96±0.03
Rand($\mathcal{T}$)	0.82±0.10	0.80±0.10	0.83±0.10	0.80±0.08
Rand($\mathcal{G}$)	0.78±0.10	0.77±0.10	0.82±0.11	0.81±0.06

obtained after 100 executions. We have measured the N_J coefficient, the Rand index w.r.t. to the correct partition that has been inferred from the literature, and the Goodman-Kruskal's coefficient to evaluate the co-clustering quality. Results are in Tab. 1.7.

There is a significant difference between the two data sets. In malaria, the average number of jumps (N_J) is already small with both algorithms. In particular, if COCLUSTER enables to get a good Goodman-Kruskal's coefficient, the co-clusters obtained by CDK-MEANS are more consistent with the biological knowledge (i.e., the partition has a higher Rand index). We notice that the number of comparisons is rather high. What we expect here, is that a constrained approach can obtain the same clustering results by using less computing resources. Instead, for drosophila, both algorithms fail in finding the correct partitioning w.r.t. the available biological knowledge. The number of jumps is in both cases high, while the Rand index is relatively low. In this case we expect to obtain better results with our constrained co-clustering approach.

We have defined the Interval constraint on the biological condition dimension. Different levels of the Max-gap constraint have been applied and we have studied the impact on the final partition by measuring the N_J coefficient, the Rand index, the Goodman-Kruskal's coefficient, and the average number of comparisons. Results are in Fig. 1.3 and Fig. 1.4, respectively for malaria and drosophila.

For malaria, the best results in terms of number of jumps (see Fig. 1.3(a)) are for a Max-gap constraint of 1 and 2. When Max-gap=2, the Rand index (Fig. 1.3(b)) is higher, and the Goodman-Kruskal's coefficient (Fig. 1.3(c)) is

TABLE 1.7: Co-clustering without interval constraints (100 trials).

	COCLUSTER			CDK-MEANS			
Dataset	N_J	*Rand*	$\tau_{\mathcal{S}}$	N_J	*Rand*	$\tau_{\mathcal{S}}$	*CC*
malaria	0.85	0.761	0.494	0.3	0.877	0.438	3.063M
drosophila	6.39	0.692	0.513	4.29	0.601	0.424	1.652M

maximum (and similar to the one obtained without constraint, see Tab. 1.7). An important observation is that the average comparison numbers (Fig. 1.3(d)) for these values of the Max-gap constraint are sensibly reduced (by a factor of 8, for Max-gap=3, up to 28 for Max-gap=2). When Max-gap is set to 1, the average comparison number is about 1/1000 of the one obtained without specifying any constraint. When Max-gap is 5, we obtain a rather bad N_J index, but the Rand coefficient is max (and similar to the one obtained without constraint). An optimal choice in this context seems to be Max-gap=2: it sensibly reduces the computational time, and it produces good clustering results. We notice also that our definition of the Max-gap constraint works for open time intervals. By setting an open time Interval constraint, we are always able to obtain a circular sequence of intervals, i.e., capturing typical developmental life cycles.

For drosophila, the improvements are more obvious. Unconstrained clustering results have shown that good partitions (with a high Goodman-Kruskal's coefficient) contain a lot of jumps. With a Max-gap constraint of 2 or 3, we can sensibly reduce the number of jumps (Fig. 1.4(a)) and it increases the quality of the partition (Fig. 1.4(b)) w.r.t. the available biological knowledge. The fact that for these Max-gap values, the Goodman-Kruskal's coefficient is minimum (Fig. 1.4(c)), indicates that the partition which better satisfies the constraints is not necessarily the "best" one. Moreover, the average number of comparisons (Fig. 1.4(d)) is reduced by 60 (Max-gap=2) and 30 (Max-gap=3).

1.4.3.1 Using Non-interval constraint

We have shown how Interval constraints can support the discovery of time interval clusters. Within some data (e.g., malaria), an unconstrained approach already gives perfect intervals, and then the question is: is it possible to discover different gene associations which hold between time points belonging to different intervals? To answer this question, we applied the Non-interval constraint to the gene expression data concerning adult time samples of the drosophila melanogaster life cycle. Indeed, time samples from t_1 to t_{10} concern the first days of male adult individual life cycle while time samples from t_{11} to t_{20} concern female individuals.

When we apply CDK-MEANS (with $k = 2$) without specifying any constraint, the two intervals $t_1, \ldots, t_{10}$ and $t_{11}, \ldots, t_{20}$ are well identified in the 100 executions. Then, we obtain almost exactly a co-cluster of males and

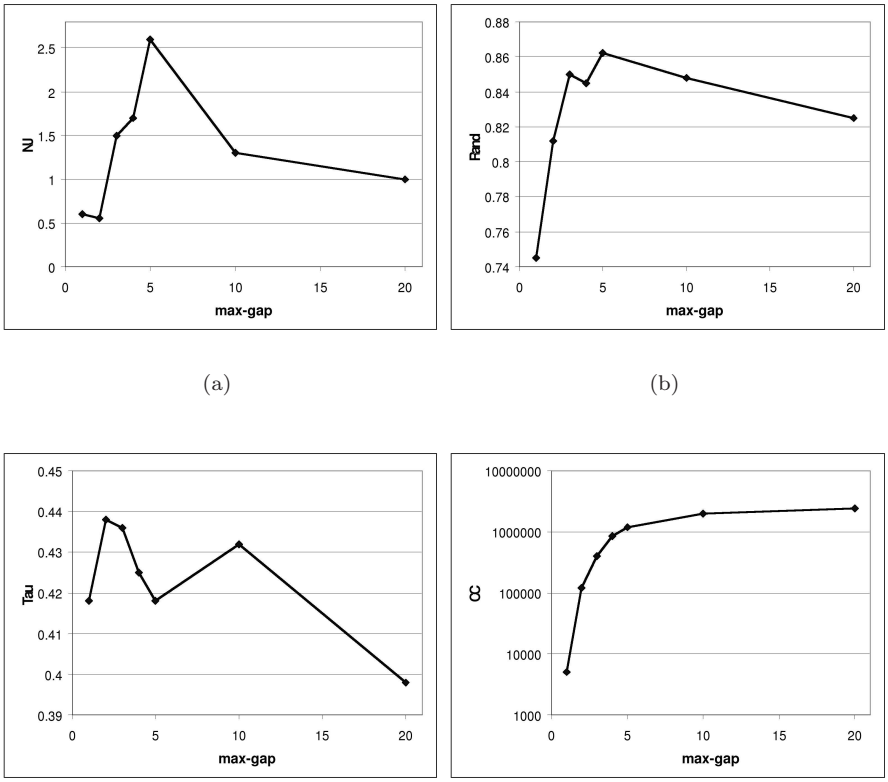
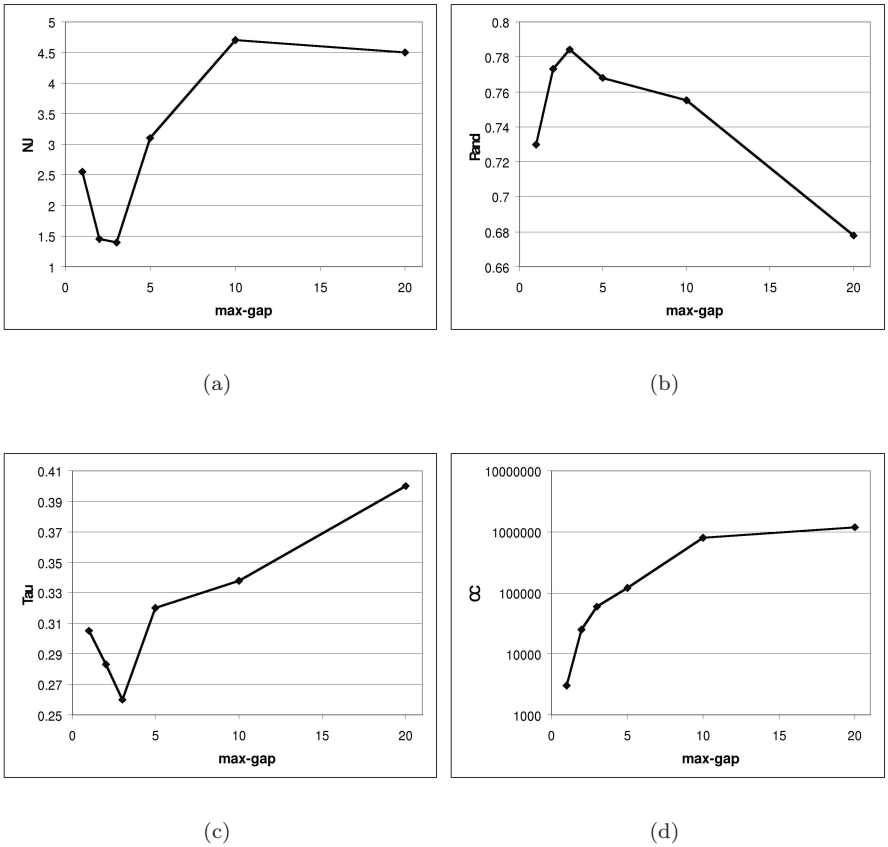


FIGURE 1.3: Jump number (a), Rand index (b), Goodman-Kruskal's coefficient (c) and comparison coefficient (d) on malaria.

a co-cluster of females and the average jump number is low. Moreover, the Goodman-Kruskal's coefficient and the loss in mutual information appears rather stable (see cdk:unconst result on Tab. 1.8). We computed these coefficients on the 100 co-clusterings returned by COCLUSTER and we noticed a significant instability (see Tab. 1.8). It seems that there are two optimum points for which the two measures are distant. For 56 runs, we got a high τ coefficient (mean 0.5605), for the other 44 ones the τ coefficient was sensibly smaller (mean 0.1156). If we consider each group of results separately, the standard deviation is significantly smaller. It means that these two results are two local optima for the COCLUSTER heuristics. Furthermore, the first group of solutions reflects the male and female repartition of the individuals, while in the second group each cluster contains both male and female individuals.



(a) (b)

(c) (d)

FIGURE 1.4: Jump number (a), Rand index (b), Goodman-Kruskal's coefficient (c) and comparison coefficient (d) on drosophila.

The average Rand value is 0.69 and the standard deviation is 23% of the mean. Also the jump number has a high and unstable value. Then, we tried to specify a Min-gap constraint on the collection of formal concepts to enforce the discovery of non Interval clusters. Even for small values of the Min-gap constraint, the average Rand value is high, while the standard deviation is lower (12% of the mean for Min-gap=2, 4% for min-gap=3) w.r.t. COCLUSTER results. The cdk:nonint row in Tab. 1.8 summarizes the more stable (w.r.t. the τ coefficient) results obtained with Min-gap=10. We also tested whether an Interval constraint could influence the stability of the co-clustering. Setting Max-gap=5 enables to get more stable co-clusterings where the Rand index is always equal to one (see cdk:int results in Tab. 1.8). These results show that, by specifying an Interval or a Non-interval constraint, the user gets

TABLE 1.8: Clustering adult drosophila individuals.

		τ		*Rand*		N_J	
bi-part.	**inst.**	**mean**	**std.dev**	**mean**	**std.dev**	**mean**	**std.dev**
co:MF	56	0.5605	0.0381	0.82	0.06	0.25	0.61
co:mixed	44	0.1156	0.0166	0.51	0.02	7.52	2.07
co:overall	100	0.3648	0.2240	0.69	0.16	3.45	3.90
cdk:unconst	100	0.4819	0.0594	0.88	0.04	1.00	0.20
cdk:int	100	0.4609	0.0347	1.00	0.00	0.00	0.00
cdk:nonint	100	0.1262	0.0761	0.53	0.04	6.94	1.93

some control on the shape of the co-clusters. An algorithm like Cocluster has sometimes found co-clusters where the sex of the individual is the major discriminative parameter. At some moment, it has captured something else. Our thesis is that a biologist might be able to have a kind of supervision on such a process. Moreover, using constraints also speeds up the co-clustering construction because we have to process a reduced collection of local patterns.

When using a co-clustering approach, it seems natural to consider both constraints on objects and on properties. In this section we have shown that even a few number of Must-link and Cannot-link constraints can support the computation of more stable co-clusterings. When a bijection exists between a partition of objects and a partition of properties (as for CDK-Means), it makes sense to set constraints which involve both objects and properties. In our framework, this extension is quite natural, and it opens new possibilities in popular applications such as document analysis and gene expression data analysis. We have also illustrated the added-value of Interval constraints. In gene expression data analysis, we often have temporal information about data. With standard (co-)clustering algorithms, there are no simple possibilities to exploit this temporal information. With a systematic approach in two different gene expression data, we were able to improve clustering results and to propose ways to control the clustering results based on domain knowledge. Here again, we can exploit such facilities for across many other application domains.

1.5 Conclusion

Co-clustering is an interesting conceptual clustering approach. Improving co-cluster relevancy remains a difficult task in real-life exploratory data analysis processes. First, it is hard to capture subjective interestingness aspects, e.g., the analyst's expectation given her/his domain knowledge. Next, when these expectations can be declaratively specified, using them during the computational process is challenging. We have shown that it was possible to use a

simple but powerful generic co-clustering framework based on local patterns. Several types of constraints on co-clusters have been considered, including new constraints when at least one of the dimensions is ordered. Applications on temporal gene expression data analysis have been sketched. Many other applications rely on ordered data analysis and might benefit from such constrained co-clustering approaches. Notice also that extended Must-link and Cannot-link constraints can be handled efficiently by our framework. A short-term perspective is to formalize the properties of the global constraints (i.e., constraints on co-clusterings) which can be, more or less automatically, transformed into local level constraints. Looking for propagation strategies that might enable to enforce Interval constraints on every computed co-cluster is also on our research agenda.

Acknowledgements. This work has been realized when Ruggero G. Pensa was a research fellow at University of Saint-Etienne (France). This research is partially funded by EU contract IQ FP6-516169 (FET arm of the IST programme).

References

[1] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo. Fast discovery of association rules. In *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, 1996.

[2] M.N. Arbeitman, E.E. Furlong, F. Imam, E. Johnson, B.H. Null, B.S. Baker, M.A. Krasnow, M.P. Scott, R.W. Davis, and K.P. White. Gene expression during the life cycle of drosophila melanogaster. *Science*, 297:2270–2275, 2002.

[3] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugua, and D. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. *Journal of Machine Learning Research*, 8:1919–1986, 2007.

[4] S. Basu, M. Bilenko, and R. J. Mooney. A probabilistic framework for semi-supervised clustering. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 59–68, Seattle, USA, 2004.

[5] C. Becquet, S. Blachon, B. Jeudy, J.-F. Boulicaut, and O. Gandrillon. Strong association rule mining for large gene expression data analysis: a case study on human SAGE data. *Genome Biology*, 12:1–16, 2002.

[6] J. Besson, C. Robardet, and J-F. Boulicaut. Mining a new fault-tolerant pattern type as an alternative to formal concept discovery. In *Proceedings of the Fourteenth International Conference on Conceptual Structures*, volume 4068 of *Lecture Notes in Computer Science*, pages 144–157, Aalborg, Denmark, July 2006. Springer.

[7] J. Besson, C. Robardet, J.-F. Boulicaut, and S. Rome. Constraint-based concept mining and its application to microarray data analysis. *Intelligent Data Analysis*, 9(1):59–82, 2005.

[8] Z. Bozdech, M. Llinás, B. Lee Pulliam, E.D. Wong, J. Zhu, and J.L. DeRisi. The transcriptome of the intraerythrocytic developmental cycle of plasmodium falciparum. *Public Library of Science Biology*, 1(1):1–16, 2003.

[9] H. Cho, I. S. Dhillon, Y. Guan, and S. Sra. Minimum sum-squared residue co-clustering of gene expression data. In *Proceedings of the 2004*

0-8493-0052-5/00/$0.00+$.50

SIAM International Conference on Data Mining, pages 114–125, Lake Buena Vista, USA, 2004.

[10] I. Davidson and S. S. Ravi. Agglomerative hierarchical clustering with constraints: Theoretical and empirical results. In *Proceedings of the Ninth European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 3721 of *Lecture Notes in Computer Science*, pages 59–70, Porto, Portugal, 2005. Springer.

[11] I. Davidson and S. S. Ravi. Clustering with constraints: Feasibility issues and the k-means algorithm. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, pages 138–149, Newport Beach, USA, 2005.

[12] L. De Raedt and A. Zimmermann. Constraint-based pattern set mining. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, Minneapolis, USA, 2007.

[13] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic coclustering. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 89–98, Washington, USA, 2003.

[14] W. D. Fisher. On grouping for maximum homogeneity. *Journal of the American Statistical Association*, 53:789–798, 1958.

[15] L. A. Goodman and W. H. Kruskal. Measures of association for cross classification. *Journal of the American Statistical Association*, 49:732–764, 1954.

[16] D. Klein, S. D. Kamvar, and C. D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 307–314, Sydney, Australia, 2002. Morgan Kaufmann.

[17] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proceedings of the Fourth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 80–86. AAAI Press, 1998.

[18] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

[19] H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery journal*, 1(3):241–258, 1997.

[20] K. Morik, J-F. Boulicaut, and A. Siebes, editors. *Local Pattern Detection, International Seminar, Dagstuhl Castle, Germany, April 12-16,*

2004, Revised Selected Papers, volume 3539 of *Lecture Notes in Computer Science*. Springer, 2005.

[21] R. G. Pensa, C. Robardet, and J.-F. Boulicaut. A bi-clustering framework for categorical data. In *Proceedings of the Ninth European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 3721 of *Lecture Notes in Artificial Intelligence*, pages 643–650, Porto, Portugal, October 2005. Springer.

[22] R. G. Pensa, C. Robardet, and J-F. Boulicaut. Towards constrained co-clustering in ordered 0/1 data sets. In *Proceedings of the Sixteenth International Symposium on Methodologies for Intelligent Systems*, volume 4203 of *Lecture Notes in Computer Science*, pages 425–434, Bari, Italy, 2006. Springer.

[23] W.M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[24] C. Robardet and F. Feschet. Efficient local search in conceptual clustering. In *Discovery Science: Proceedings of the Fourth International Conference DS 2001*, volume 2226 of *Lecture Notes in Computer Science*, pages 323–335, Washington, USA, november 2001. Springer.

[25] M. Steinbach, P-N. Tan, and V. Kumar. Support envelopes: a technique for exploring the structure of association patterns. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 296–305, Seattle, USA, 2004.

[26] K. Wagstaff. Value, cost, and sharing: Open issues in constrained clustering. In *Knowledge Discovery in Inductive Databases: the Fifth International Workshop, KDID 2006 Berlin, Germany, September 18, 2006 Revised Selected and Invited Papers*, volume 4747 of *Lecture Notes in Computer Science*, pages 1–10. Springer, 2007.

[27] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 577–584, Williamstown, USA, 2001. Morgan Kaufmann.