# Toward Improving b-Coloring based Clustering using a Greedy re-Coloring Algorithm

Tetsuya Yoshida[1], Haytham Elghazel[2], Véronique Deslandres[2], Mohand-Said Hacid[3] and Alain Dussauchoy[2]
[1] *Graduate School of Information Science and Technology, Hokkaido University,*
[2] *Université de Lyon, Lyon, F-69003, France ; université Lyon 1, EA4125, LIESP*
[3] *Université de Lyon, Lyon, F-69003, France ; université Lyon 1, LIRIS,*
[1]*Japan*
[2,3]*France*

## 1. Introduction

Clustering is an important task in the process of data analysis which can be viewed as a data modeling technique that provides an attractive mechanism to automatically find the hidden structure of large data sets (Jain et al., 1999). Informally, this task consists of the division of data items (objects, instances, etc.) into groups or categories, such that all objects in the same group are similar to each other, while dissimilar from objects in the other groups. Clustering plays an important role in data mining applications such as Web analysis, information retrieval, medical diagnosis, and many other domains.

Recently, we have proposed a clustering method based on the concept of b-coloring of a graph (Irving & Manlov, 1999). A graph b-coloring is an assignment of colors to the vertices of the graph such that:

i. no two adjacent vertices (vertices joined by an weighted edge representing the dissimilarity between objects) have the same color (*proper coloring*)
ii. for each color, there exists at least one vertex which is adjacent (has a sufficient dissimilarity degree) to all other colors. This vertex is called a *dominating vertex*; there can be many within the same class.

Both (i) and (ii) are the constraints in b-coloring of a graph.

The b-coloring based clustering method enables to build a fine partition of the dataset into clusters even when the number of clusters is not specified in advance. The previous clustering algorithm in (Elghazel et al., 2006) conducts the following two steps in greedy fashion:

1. initalizes the colors of vertices so that the colors satisfy proper coloring, and
2. removes, by a greedy procedure, the colors that have no dominating vertices, until each color has at least one dominating vertex.

These steps correspond to the above two constraints in b-coloring. Although it returns a b-coloring of a graph, it does not explicitly consider the quality of the clusters in the algorithm. Thus, besides satisfying the above constraints, it was difficult to explicitly generate *better* clusters of the given data items.

Source: Advances in Greedy Algorithms, Book edited by: Witold Bednorz,
ISBN 978-953-7619-27-5, pp. 586, November 2008, I-Tech, Vienna, Austria

In order to alleviate this weakness, we have proposed a greedy algorithm which realizes the re-coloring of data items (vertices) to improve the quality of the constructed partition (Elghazel et al., 2007). Informally, our algorithm selects at each stage the vertex with the maximum degree of "outlier" and which do not affects the dominant vertices in the b-coloring. The color of the selected vertex is changed while guaranteeing that the quality of the re-colored partition is monotonically improved. The selection of vertices as well as that of the assigned colors are conducted in greedy fashion. Our greedy algorithm exhibits the following characteristics:

1. it realizes the update of b-coloring based clustering while satisfying the constraints in b-coloring,
2. it monotonically increases the quality of clusters (the quality clusters needs to be measured by some objective function). This enables to realize a compromise between the intra-cluster cohesion and intercluster separation, and
3. it employs a simple greedy strategy, in order to reduce its time complexity.

Thus, the proposed greedy algorithm can complement the weakness of the previous method by improving the constructed partition.

To evaluate the effectiveness of the proposed greedy algorithm, we tested it over benchmark datasets from the UCI repository (Blake & Merz, 1998). The detailed results of the evaluations are reported and discussed in this paper. Through this evaluation, the effectiveness of the proposed greedy algorithm is confirmed.

This paper is organized as follows. Section 2 presents the related work. Section 3 explains the approach of b-coloring based clustering and validity indices for estimating the quality of clustering in general. Section 4 describes the details of the proposed greedy algorithm. Section 5 reports the results of the experiments to evaluate the proposed algorithm. Section 6 discusses our approach in terms of the greedy strategy and other possible improvements. Section 7 gives a brief conclusion.

## 2. Related work

Generally speaking, clustering of data can be divided into two approaches: a hierarchical approach and a partitioning approach. The hierarchical approach builds a cluster hierarchy, or a tree of clusters (which is called a dendrogram) whose leaves are the data points and whose internal nodes represent nested clusters of various sizes (Guha et al., 1998). On the other hand, the partitioning approach give a single partition of the data by fixing some parameters (number of clusters, thresholds, etc.). Each cluster is represented by its centroid (Hartigan & Wong, 1979) or by one of its objects located near its center (e.g., monoid) (Ng & and Han, 2002). When the distances (or, dissimilarities) among all the pairs of data can be estimated, these can be represented as a weighed dissimilarity matrix in which each element stores the corresponding dissimilarity. Based on the dissimilarity matrix, the data can also be conceived as a graph where each vertex corresponds to a data item and each edge corresponds to a pair of data items with their dissimilarity as its label.

Other techniques for realizing the clustering of data include graph-theoretic clustering approaches. Many graph-theoretic clustering algorithms basically consist of searching for certain combinatorial structures in the similarity graph. In this case, some hierarchical approaches are related to graph-theoretic clustering. The best-known graph-theoretic divisive clustering algorithm (the single-link algorithm) is based on construction of the Minimal Spanning Tree (MST) of the data (Zahn, 1971), and then deleting the MST edges

with the largest lengths to generate single-link clusters. The complete-link algorithms are also reduced to a search for a maximal complete subgraph, namely a clique which is the strictest definition of a cluster. Some authors have proposed to use the vertex coloring of graphs for the hierarchical classification purpose. (Guenoche et al, 1991) proposed a divisive classification method based on dissimilarity tables, where the iterative algorithm consists, at each step, in finding a partition by subdividing the cluster with the largest diameter into two clusters in order to exhibit a new partition with the minimal diameter. By mapping each data item to the corresponding vertex, the subdivision is obtained by a 2-coloring of the vertices of the maximum spanning tree built from the dissimilarity table. The derived classification structure is a hierarchy.
On the other hand, the partitioning methods are also related to graph-theoretic clustering. The method in (Hansen & Delattre, 1978) reduced the partitioning problem of a data set into clusters with minimal diameter, to the minimal coloring problem of a superior threshold graph. The edges of this graph are the pairs of vertices distanced from more than a given threshold. In such a graph, each color corresponds to one cluster and the number of colors is minimal. Unfortunately, while this method tends to build a partition of the data set with effectively compact clusters, it does not give any importance to the cluster-separation.

## 3. b-coloring based clustering

We use a bold italic capital letter to denote a set. For instance, $\boldsymbol{V}$ represents a set of vertices. In addition, $|\boldsymbol{V}|$ represents the cardinality of $\boldsymbol{V}$, i.e., the number of vertices in $\boldsymbol{V}$.
Our approach for the clustering of data assumes that some dissimilarity function for a pair of data to be handled is specified. By denoting the set of data to be handled as $\boldsymbol{V}$, the dissimilarity of a pairs of data $v_i, v_j \in \boldsymbol{V}$ is calculated by some function $d$: $\boldsymbol{V} \times \boldsymbol{V} \rightarrow \boldsymbol{R}^+$. We also assume that this function is symmetric.
Based on the dissimilarity function, the set of data $\boldsymbol{V}$ can be transformed into the corresponding graph-structured data by:

1. mapping each data to a vertex, and
2. connecting each pair of vertices $v_i$ and $v_j \in \boldsymbol{V}$ by the edge $(v_i, v_j)$ with label $d(v_i, v_j)$.

The above transformation results in an undirected complete edge-weighted graph. The b-coloring of this complete graph is not interesting in terms of the clustering problem. Indeed, each data will be assigned to one and the only one cluster, which is meaningless as the clustering of data. To alleviate this, we also require another parameter $\theta$. This parameter works as the threshold value for defining the edges in the graph. Formally, a pair of vertices $(v_i, v_j \in \boldsymbol{V})$ are connected with the edge $(v_i, v_j)$ in the graph iff $d(v_i, v_j) > \theta$ for the specified threshold $\theta$. The constructed graph $G(\boldsymbol{V},\boldsymbol{E})$ is called a superior threshold graph.
The above notations are summarized in Table 1.

| Symbol | Description |
|---|---|
| $\boldsymbol{V}$ | a set of vertices (each vertex corresponds to a data item) |
| $\theta$ | a threshold value |
| $\boldsymbol{E}$ | the set of edges among $\boldsymbol{V}$ for $d(,)$ and $\theta$ |
| $\boldsymbol{P}$ | a set of clusters |
| $d(v_i, v_j)$ | a dissimilarity function between vertices $v_i$ and $v_j$ |

Table 1. Notations for a threshold graph

### 3.1 An example

Suppose a set of data with the dissimilarities in Table 2 is given, which is represented as a dissimilarity matrix for the data. Fig. 1 shows the superior threshold graph for Table 2 where the threshold $\theta$ is set to 0.15. The edges are labeled with the corresponding dissimilarities in the matrix.

| vertex | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| A | 0 | | | | | | | | |
| B | 0.20 | 0 | | | | | | | |
| C | 0.10 | 0.30 | 0 | | | | | | |
| D | 0.10 | 0.20 | 0.25 | 0 | | | | | |
| E | 0.20 | 0.20 | 0.15 | 0.40 | 0 | | | | |
| F | 0.20 | 0.20 | 0.20 | 0.25 | 0.65 | 0 | | | |
| G | 0.15 | 0.10 | 0.15 | 0.10 | 0.10 | 0.75 | 0 | | |
| H | 0.10 | 0.20 | 0.10 | 0.10 | 0.05 | 0.05 | 0.05 | | |
| I | 0.40 | 0.075 | 0.15 | 0.15 | 0.15 | 0.15 | 0.15 | 0.15 | 0 |

Table 2. A weighted dissimilarity matrix

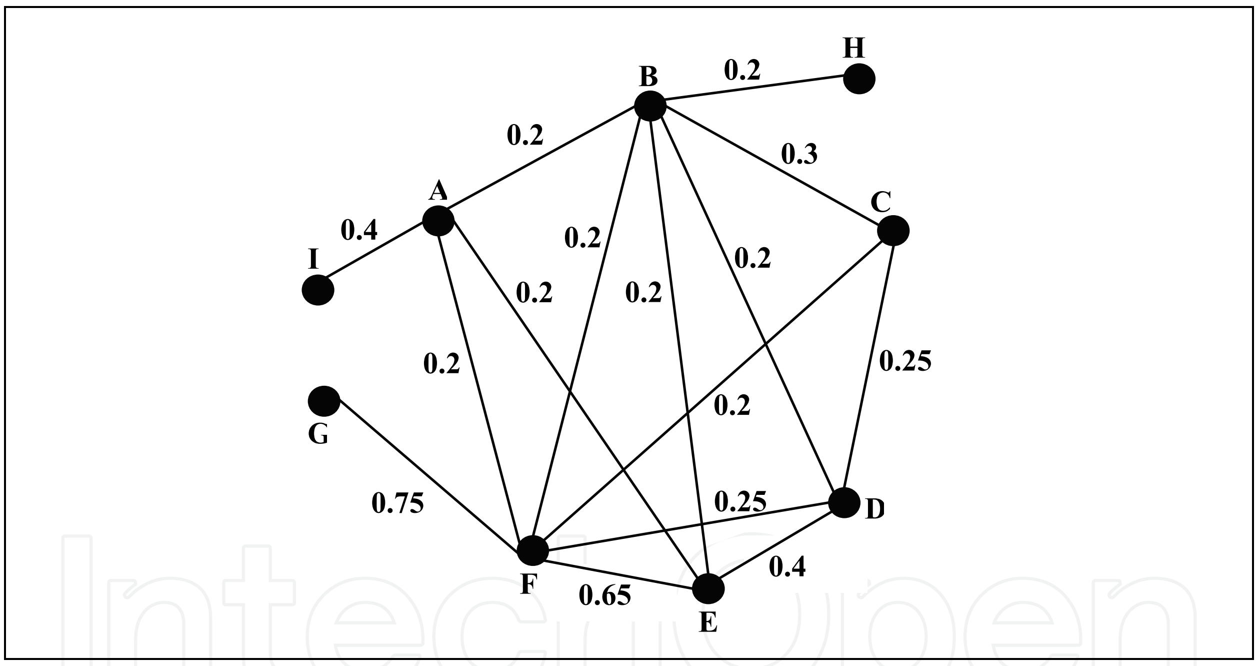

Fig. 1. A threshold graph for Table 1 ($\theta$=0.15)

The previous b-coloring based clustering algorithm works with the following two stages:

1. initializing the colors of vertices so that the colors satisfy proper coloring, and
2. removing, by a greedy procedure, the colors without any dominating vertex.

Here, these stages are conducted with greedy fashion, because finding the maximum number of colors for b-coloring of a graph is known to be computationally too expensive. Utilization of a greedy strategy is a realistic approach for dealing with real-world data, especially for large scale data.

For instance, the proper coloring in Fig.2 is obtained from the graph in Fig.1 with step 1). After that, a b-coloring of the graph is obtained by step 2). The result is illustrated in Fig. 3. The vertices with the same color (shape) are grouped into the same cluster. This realizes the clustering of data in Table 1. In this example, the sets of vertices {A,D}, {B}, {C,E,G,I}, {F} are the clusters.

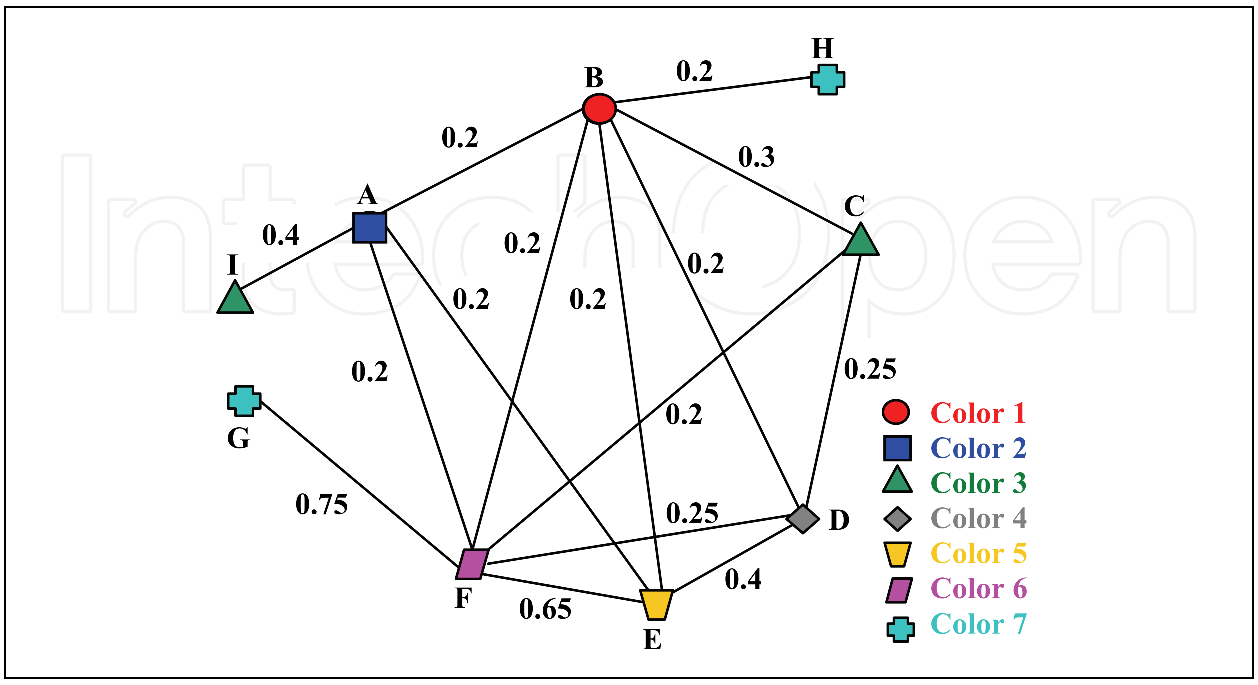

Fig. 2. A proper coloring of the graph in Fig.1

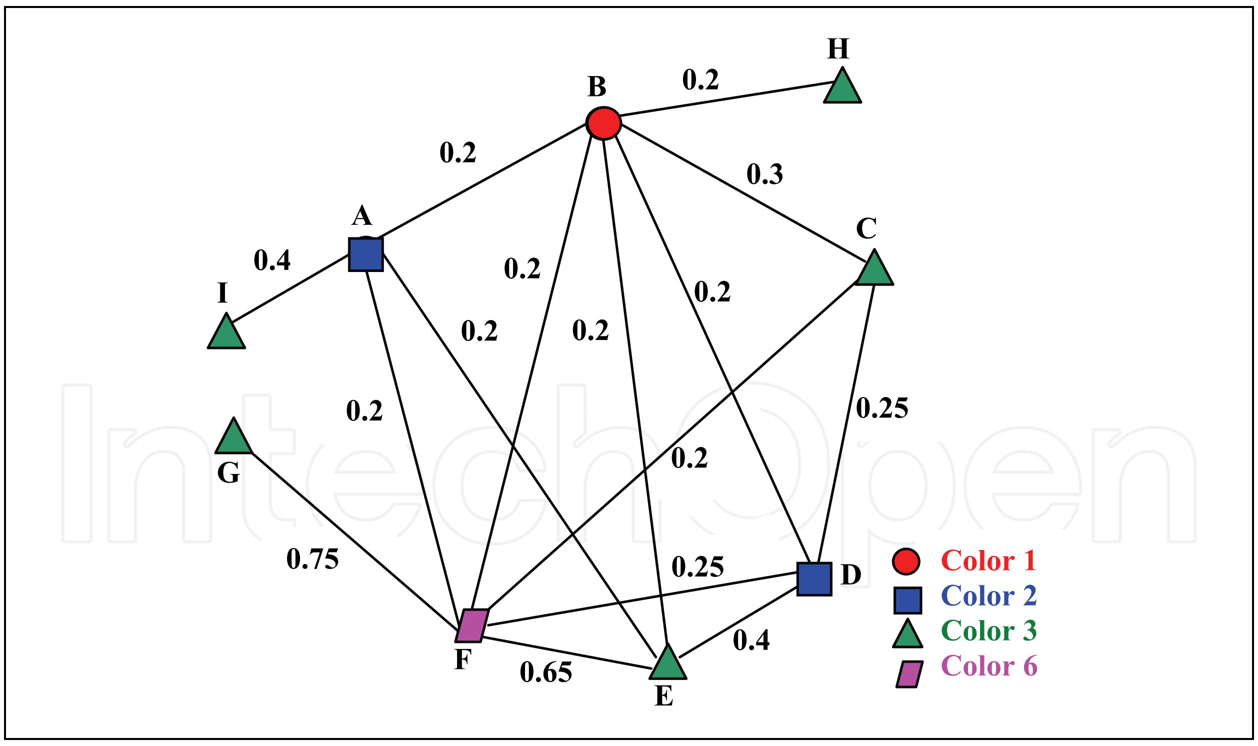

Fig. 3. A b-coloring of the graph in Fig.1 based on Fig.2.

### 3.2 Validation Indices

Many validation indices for clustering have been proposed (Bezdek & Pal, 1998) and adapted to the symbolic framework (Kalyani & Sushmit, 2003). Among them, we focus on a

validation index called generalized Dunn's index. This index is denoted as $Dunn_G$ hereafter in this paper. This index is designed to offer a compromise between the *inter-cluster separation* and the *intra-cluster cohesion*. The former corresponds to the compactness of the clusters, the latter corresponds to what extent the clusters are well-separated each other.
Suppose a set of vertices $\boldsymbol{V}$ (which correspond to the data items) are clustered or grouped into a partition $\boldsymbol{P} = \{C_1, C_2, \ldots, C_k\}$. Here, each cluster or group is denoted as $C_i$, and the partition $\boldsymbol{P}$ satisfies the constraint: $\forall C_i, C_j \in \boldsymbol{P}$, $C_i \cap C_j = \varnothing$ for $i \neq j$. We abuse the notation of $\boldsymbol{P}$ to represent both a set of clusters as well as a set of colors, because each cluster $C_i \in \boldsymbol{P}$ corresponds to a color in our approach and no cluster share the same color.
For $\forall C_h \in \boldsymbol{P}$, an average within-cluster dissimilarity is defined as

$$S_a(C_h) = \frac{1}{\eta_h(\eta_h - 1)} \sum_{o=1}^{\eta_h} \sum_{o'=1}^{\eta_h} d(v_o, v_{o'}) \tag{1}$$

where $\eta_h = |C_h|$, $v_o, v_{o'} \in C_h$.
For $\forall C_i, C_j \in \boldsymbol{P}$, an average between-cluster dissimilarity is defined as

$$d_a(C_i, C_j) = \frac{1}{\eta_i \eta_j} \sum_{p=1}^{\eta_i} \sum_{q=1}^{\eta_j} d(v_p, v_q) \tag{2}$$

where $\eta_i = |C_i|$ and $\eta_j = |C_j|$, $v_p \in C_i$, $v_q \in C_j$.
Dunn's generalized index for a partition $P$ is defined as

$$Dunn_G(P) = \frac{\min_{i,j,i \neq j} d_a(C_i, C_j)}{\max_h S_a(C_h)} \tag{3}$$

where $C_h$, $C_i$, $C_j \in \boldsymbol{P}$.
Basically, the partition $\boldsymbol{P}$ with the largest $Dunn_G$ ($\boldsymbol{P}$) is regarded as the best clustering.
The above notations are summarized in Table 3.

| Symbol | Description |
|---|---|
| $S_a(C_h)$ | an average within-cluster dissimilarity of a cluster $C_h$ |
| $d_a(C_i, C_j)$ | an average between-cluster dissimilarity between $C_i$ and $C_j$ |
| $Dunn_G$ ($\boldsymbol{P}$) | generalized Dunn's index of a partition $P$ |

Table 3. Notations for evaluating a partition.

## 4. A greedy re-coloring algorithm

### 4.1 A motivating example

As explained in Section 3, for the data in Table 2, the previous approach returns the partition in Fig. 3 as its best b-coloring of the corresponding superior threshold graph. However, even for the same number of clusters, the graph in Fig. 1 can have other different b-colorings with better quality of clustering *(e.g.,* with larger value of $Dunn_G$ index). Actually, there is another b-coloring with better quality of clusters. An example is shown in Fig. 4. Obviously, the colors in Fig.4 satisfy the constraints in b-coloring and thus it is a b-coloring of the graph in Fig.1. Furthermore, the partition in Fig. 4 is better than that in Fig.3, since it has the value $Dunn_G$ =1.538, which is larger than the previous value (1.522) in Fig. 3.

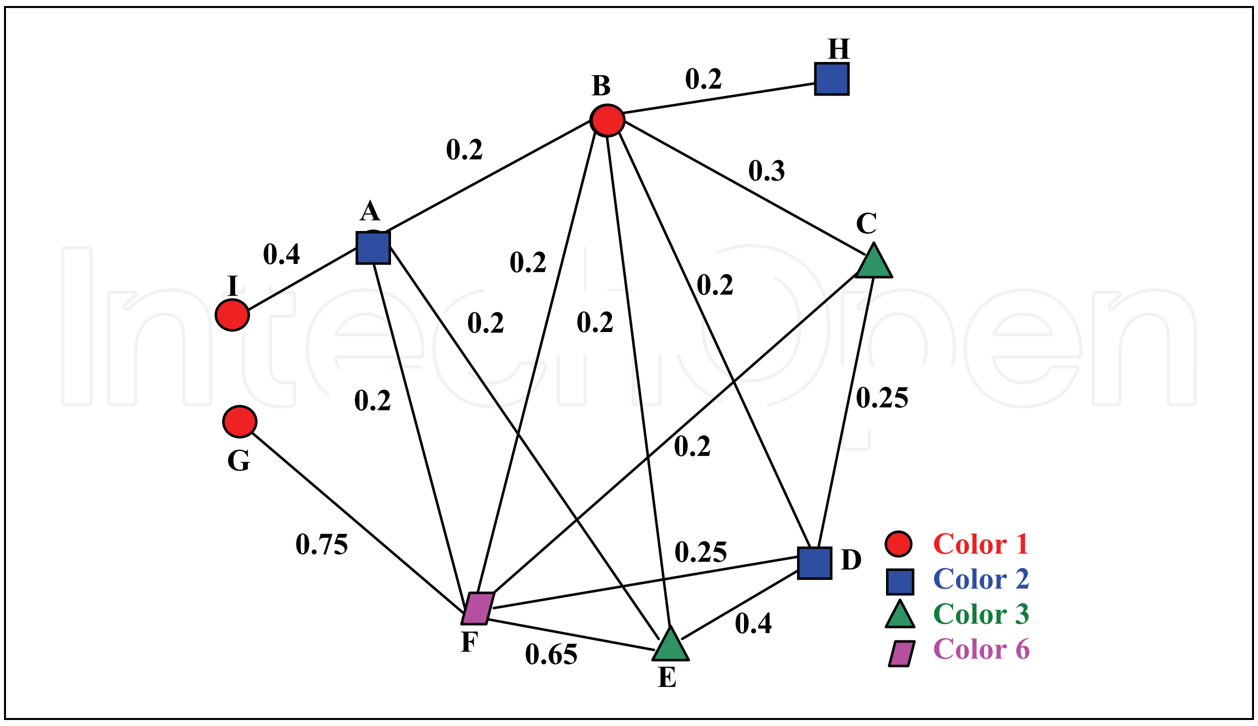

Fig. 4. Another b-coloring with better quality

As illustrated in the above example, even when the previous approach described in Section 3 returns a partition ***P*** based on the b-coloring of a given graph $G(\boldsymbol{V},\boldsymbol{E})$, there can be other partitions for the same graph with better quality, while satisfying the constraints in b-coloring. To construct a better partition, it is also important to find a partition with better quality, while satisfying the constraints b-coloring.

However, as described above, *directly* trying to find out a better b-coloring can be computationally too expensive. Even if a better partition can be obtained, it will not scale up for large data. To cope with this problem, we take the following approach: instead of directly finding out a better partition, by utilizing a partition which satisfies the constraints, try to find out better ones. This approach is formalized as follows.

**[Definition 1] Re-Coloring Problem in b-Coloring based Clustering**

For a given graph $G(\boldsymbol{V},\boldsymbol{E})$ and a b-coloring partition ***P*** of $G(\boldsymbol{V},\boldsymbol{E})$, find another b-coloring partition ***P′*** of $G(\boldsymbol{V},\boldsymbol{E})$ such that ***P′*** is equal to or better than ***P*** for some clustering validity index.

In our current approach, the quality of a partition ***P*** is measured with $Dunn_G(\boldsymbol{P})$ in Section 3.2. In the following, we describe the details of our approach to tackle this problem.

## 4.2 Notations

In addition to the notations in Table3, to characterize a vertex $v \in \boldsymbol{V}$ in a graph $G(\boldsymbol{V},\boldsymbol{E})$, we use the following functions for in the description of our greedy algorithm. A function $N(v)$ returns the set of neighboring vertices in $G(\boldsymbol{V},\boldsymbol{E})$. A function $c(v)$ returns the assigned color of the vertex $v$. A function $N_c(v)$ returns the set of neighborhood colors for the vertex $v$. Furthermore, a function $C_p(v)$ is defined as $C_p(v) = \boldsymbol{P} \backslash N_c(v)$ for $v$. Here, $\boldsymbol{P} \backslash N_c(v)$ represents the set difference. Note that $C_p(v)$ contains the originally assigned color $c(v)$ of the vertex $v$. These are summarized in Table 4.

| Symbol | Description |
|---|---|
| $N(v)$ | the set of neighborhood vertices for a vertex $v \in \boldsymbol{V}$ in $G(\boldsymbol{V},\boldsymbol{E})$ |
| $c(v)$ | the assigned color of a vertex $v \in \boldsymbol{V}$ in $G(\boldsymbol{V},\boldsymbol{E})$ |
| $N_c(v)$ | the set of neighborhood colors for a vertex $v \in \boldsymbol{V}$ in $G(\boldsymbol{V},\boldsymbol{E})$ |
| $C_p(v)$ | $C_p(v) = \boldsymbol{P} \setminus N_c(v)$ for a vertex $v \in \boldsymbol{V}$ in $G(\boldsymbol{V},\boldsymbol{E})$ |

Table 4. Several functions for a vertex in a graph

### 4.2.1 Types of vertex

A set of vertices $\boldsymbol{V_d}$ contain the dominating vertices in a b-coloring of a graph $G(\boldsymbol{V},\boldsymbol{E})$. For each vertex $v \in \boldsymbol{V_d}$, if a vertex $v_s$ is the only vertex with the color $c(v_s)$ in $N_c(v)$, $v_s$ is called a supporting vertex of $v$.
We divide the set of vertices $\boldsymbol{V}$ into two disjoint subsets $\boldsymbol{V_c}$ and $\boldsymbol{V_{nc}}$ such that $\boldsymbol{V_c} \cup \boldsymbol{V_{nc}} = \boldsymbol{V}$ and $\boldsymbol{V_c} \cap \boldsymbol{V_{nc}} = \emptyset$. Each vertex in $\boldsymbol{V_c}$ is called a critical vertex, and each vertex in $\boldsymbol{V_{nc}}$ is called a non-critical vertex. The vertices in $\boldsymbol{V_c}$ are critical in the sense that their colors cannot be changed (or, would not be changed in our current approach). On the other hand, the vertices in $\boldsymbol{V_{nc}}$ are not critical and thus considered as the candidates for the re-coloring. $\boldsymbol{V_c}$ is further divided into three disjoint sets of vertices $\boldsymbol{V_d} \cup \boldsymbol{V_s} \cup \boldsymbol{V_f}$ . Here, $\boldsymbol{V_f}$ is called a set of finished vertices, and contains the already checked vertices for re-coloring. More detailed discussions about why it is important to define these sets of vertices are given in Section 4.3, with respect to the greedy nature of our algorithm. Furthermore, for $\forall v \in \boldsymbol{V}$, $\forall C_i \in \boldsymbol{P}$, an average dissimilarity between a vertex $v$ and a cluster $C_i$ is defined:

$$d_a(v, C_i) = \frac{1}{\eta_i} \sum_{p=1}^{\eta_i} d(v, v_p) \tag{4}$$

where $\eta_i = |C_i| v_p \in C_i$.
The above notations are summarized in Table 5.

| Symbol | Description |
|---|---|
| $\boldsymbol{V_c}$ | the set of critical vertices |
| $\boldsymbol{V_{nc}}$ | the set of non-critical vertices |
| $\boldsymbol{V_d}$ | the set of dominating vertices |
| $\boldsymbol{V_s}$ | the set of supporting vertices |
| $\boldsymbol{V_f}$ | the a set of finished vertices |
| $d_a(v, C_i)$ | an average dissimilarity between a vertex $v$ and a cluster $C_i$ |

Table 5. Notations for the vertices in a graph

## 4.3 A greedy re-coloring algorithm

### 4.3.1 Why greedy algorithm?

By definition, since each dominating vertex is connected to the vertices with all the other colors (clusters), it is far away from the other clusters (at least greater than the specified threshold $\theta$. This means that, dominating vertices contribute to increase the inter-cluster dissimilarity, which is important for better clustering.
Likewise, by definition, each supporting vertices is necessary (important) to ``keep'' some dominating vertex, since without its color the dominance property will be lost. Thus, these vertices also contributes to increase the inter-cluster dissimilarity.

Based on the above reasons, in our current approach, the colors of the vertices in $\boldsymbol{V_d}$ and $\boldsymbol{V_s}$ are fixed (not changed) to sustain the inter-cluster dissimilarity. Furthermore, to guarantee the termination of the processing, re-coloring of vertices is tried at most once. To realize this, when a vertex is tested (checked) for re-coloring, the vertex is moved into the finished vertices $\boldsymbol{V_f}$ in order to avoid the repetition.
In summary, we consider re-coloring of the vertices in $\boldsymbol{V} \setminus \{\boldsymbol{V_d} \cup \boldsymbol{V_s} \cup \boldsymbol{V_s}\}$, namely the set of non-critical vertices $\boldsymbol{V_{nc}}$. In addition, whenever a vertex is checked for re-coloring, it is moved into the finished vertices $\boldsymbol{V_f}$ so that its color is fixed in the latter processing. Thus, since the size $|\boldsymbol{V_{nc}}|$ is monotonically decreased at each re-coloring of some vertex in $G(\boldsymbol{V},\boldsymbol{E})$, the termination of the processing is guaranteed.
Since the color of a vertex $v$ is fixed once it is inserted into $\boldsymbol{V_c}$, and other possibilities are not explored in later processing, our algorithm works in a greedy fashion. This is important, both for reducing the time complexity of the algorithm and to guarantee its termination. Admittedly there can be other approach to solve the re-coloring problem. For instance, it might be possible to incorporate some kind of back-tracking for the re-coloring, *e.g.*, to consider further re-coloing of the vertices in $\boldsymbol{V_f}$. However, in compensation for the possibly better quality, such an approach will require much more computation time and more dedicated mechanism to guarantee the termination.

### 4.3.2 A vertex selection criterion

As explained in Section 4.3.1, our approach considers the vertices in $\boldsymbol{V_{nc}}$ for re-coloing. The next question is, which vertices should be considered for re-coloring and in what order. Our criterion for the vertex selection is as follows.
Among the vertices in $\boldsymbol{V_{nc}}$, we select the vertex with the maximal average within-cluster dissimilarity. Thus, the following vertex $v^*$ is selected.

$$v^* = \arg\max_{v \in V_{nc}} d_a(v, c(v)) \tag{5}$$

Here, the value of $d_a(v,c(v))$ defined in equation (4) corresponds to the degree of "outlier" of the vertex $v$, because it represents the average within-cluster dissimilarity when it is assigned to the cluster $c(v)$ (note that a color also corresponds to a cluster).
On the contrary, suppose other vertex $v'$ which is not with the maximal value in equation (4) is selected and re-colored. In that case, the size of the cluster $|C_p(v)|$ can decrease, since some other vertex might be moved into the set of critical vertices $\boldsymbol{V_c}$ due to the re-coloring of $v'$. This amounts to putting more constraints into the re-coloring processing and restricting the possibilities of new color for $v^*$. For instance, the increase in the size of neighboring vertices $|N_c(v^*)|$ means more constraints, and thus leads to decreasing the possible colors for $v^*$.
Based on the above argument, among the vertices in $\boldsymbol{V_{nc}}$, we select the vertex with the maximal average within-cluster dissimilarity for re-coloring.

### 4.3.3 A color selection criterion

After selecting a vertex as the candidate for re-coloring, the next question is, which color the vertex should be assigned. Note that our objective is to increase the quality of a partition, while preserving the constraints. The second constraint, namely the preservation of the dominating vertices is guaranteed by our vertex selection strategy in Section 4.3.1. Thus, we need to select the color which satisfy the first constraint, namely the proper coloring, and which leads to the better quality of the resulting partition.

Our color selection criterion is as follows. When the vertex $v^*$ is selected for re-coloring, we check the colors in $C_p(v^*)$, since it represents the colors which satisfy the proper coloring constraint for $v^*$. Among these colors, we select the one with the maximal $Dunn_G$ in equation (3), since it evaluates the quality of the resulting partition.

### 4.3.4 The algorithm

We need to take into account the fact that the color of non-critical vertices $\boldsymbol{V_{nc}}$ might be changed through their re-coloring in the latter processing. This means that, reflecting the colors of $\boldsymbol{V_{nc}}$ to evaluate the quality of the current partition can be an unreliable. Thus, we exclude the non-critical vertices to evaluate the quality of the current partition in the re-coloring process, and utilize only the fixed colors in critical vertices $\boldsymbol{V_c}$.

Furthermore, when the color $c(v)$ of a vertex $v$ is re-colored to some other color $c$, some vertex $v_{nc} \in \boldsymbol{V_{nc}}$ might become new critical vertices. This is because some other vertices can become dominating vertices or supporting ones, due to the re-coloring of $v$. To reflect the change of colors in the graph $G(\boldsymbol{V},\boldsymbol{E})$ due to the re-coloring of the vertex $v$, we also define a set of vertices $\boldsymbol{V_c}^{tmp}(v, c)$. $\boldsymbol{V_c}^{tmp}(v, c)$ represents the set of vertices which become *new* critical vertices induced from this re-coloring. In addition, we denote the resulting partition of $G(\boldsymbol{V},\boldsymbol{E})$ as $\boldsymbol{P}(v, c)$. Here, in $\boldsymbol{P}(v, c)$, only the originally assigned color $c(v)$ of the vertex $v$ is re-colored to $c$, and the colors of the other remaining vertices are not changed. These notations are summarized in Table 6.

| Symbol | Description |
|---|---|
| $\boldsymbol{V_c}^{tmp}(v, c)$ | the a set of new critical vertices by changing the color of $v$ to $c$ |
| $\boldsymbol{P}(v, c)$ | the new partition by changing the color of $v$ to $c$ |

Table 6. additional notations for the algorithm

Based on the above, our greedy re-coloring algorithm is summarized as the Algorithm re-coloring() in Fig. 5. In the selection of vertex or color, there can be multiple candidates with exactly the same value. In that case, since the candidates are indistinguishable with respect to our criteria, one of them is selected at random.

### 4.3.5 Properties of the algorithm

The proposed algorithm has the following desirable properties for clustering. We explain the properties with their proofs in this subsection.

**[Proposition 1]**

Algorithm re-coloring() returns a proper coloring of $G(\boldsymbol{V},\boldsymbol{E})$ from $\boldsymbol{P}$.

**Proof**

Algorithm re-coloring() can change the color of a vertex $v^*$ only to some color in $C_p(v^*)$. By definition of the function $C_p()$ in Table, all the colors in $C_p(v^*)$ satisfy the proper coloring for the vertex $v^*$.

**[Proposition 2]**

Algorithm re-coloring() returns a b-coloring of $G(\boldsymbol{V},\boldsymbol{E})$ from $\boldsymbol{P}$.

**Proof**

From Proposition 1, proper coloring is guaranteed. We need to show that there is at least one dominating vertex for each color. By definition, this property is satisfied in $\boldsymbol{P}$. As explained in Section 4.3.3, since Algorithm re-coloring() does not change the colors of dominating vertices nor those of the supporting vertices, there is at least one dominating vertex for each color.

```
Algorithm re_coloring()
Input:  G(V,E) //A graph
        P      //a b-coloring partition of G(V,E)
Output: C'     //another partition

C' := P;
Divide $V$ into $V_c \cup V_{nc}$
while $V_{nc} \neq \emptyset$ do
    $v^* = \arg\max_{v \in V_{nc}} d_a(v', c(v'))$
    for $c \in C_p(v^*)$ do
        calculate $V_c^{tmp}(v^*, c)$ and $P(v^*, c)$ induced from the re-coloring of $c(v^*)$ into $c$;
        For $\forall\ C_i \in P(v^*, c)$, calculate $d_a(v^*, C_i))$ w.r.t. $V_c \cup V_c^{tmp}(v^*, c)$
    end for
    $c^* = \arg\max_{c \in C_p(v^*)} Dunn_G(P(v^*, c))$
    Re-color $c(v^*)$ into $c^*$;
    $V_{nc} = V_{nc} \setminus \{v^*\};;$
    $V_f = V_f \cup \{v^*\};$
    $V_c = V_c \cup V_c^{tmp}(v^*, c);$
end while
return C'
```

Fig.5. the greedy re-coloring algorithm

**[Proposition 3]**
Algorithm re-coloring() monotonically improve the quality of partition.
**Proof**
As explained above, the color which maximizes the quality (here, $Dunn_G$ is utilized)[ is selected by modifying the originally assigned color. Note that it is allowed that the originally assigned color is selected and thus unchanged. Since this re-coloring is repeated for all the non-critical vertices, when Algorithm re-coloring() terminates, the quality of partition will be monotonically improved.

## 5. Evaluations

The proposed greedy algorithm (Algorithm re-coloring() in Fig.) was tested by considering two relevant benchmark data sets, viz., *Zoo*, and *Mushroom* from the UCI Machine Learning Repository (Blake & Merz, 1998). To evaluate the quality of the partition discovered by the greedy algorithm (called Improved b-coloring Partition), the results are compared with that of the best partition returned by the previous b-coloring clustering algorithm as the one maximizing the $Dunn_G$ value (denoted as original b-coloring), the Hansen's method based on minimal coloring technique (Hansen & Delattre, 1978) and the Agglomerative Single-link method (Jain et al., 1999).
In addition to the value of Generalized Dunn's index, we also evaluated the results based on a probability matching scheme called *Distinctness* (Kalyani & Sushmita, 2003). This evaluation index is useful in the cluster validation problem, since it is independent of

1. the number of clusters, and
2. the dissimilarity between objects.

For a partition ***P*** with *p* clusters $\{C_1, C_2, .., C_p\}$, the *Distinctness* is defined as the inter-cluster dissimilarity using a probability match measure, namely the variance of the distribution match. The variance of the distribution match between clusters $C_k$ and $C_l$ in a given partition is measured as:

$$Var(C_k, C_l) = \frac{1}{m}\sum_{i}^{m}\sum_{j}\left(P\left(a_i = V_{ij}|\mathrm{C_k}\right) - P\left(a_i = V_{ij}|\mathrm{C_l}\right)\right)^2 \tag{6}$$

where $m$ is the number of attributes $a_i$ characterizing the objects. $P(a_i=V_{ij} \mid C_k)$ is the conditional probability of $a_i$ to take value $V_{ij}$ in class $C_k$.

The above equation assumes that each data has only one value per attribute (represented by $j \in a_i$). The greater this value, the more dissimilar are the two clusters being compared. Thus, the concepts they represent are also dissimilar.

The *Distinctness* of a partition ***P*** is calculated as the average variance between clusters as:

$$Distinctness = \frac{\sum_{k=1}^{p}\sum_{l=1}^{p} Var(C_k, C_l)}{p(p-1)} \tag{7}$$

When comparing two partitions, the one with larger distinctness would be considered as better one, with respect to this index, since the clusters in such a partition represent more distinct concepts.

### 5.1 Zoo dataset

The Zoo dataset includes 100 instances of animals with 17 features and 7 output classes. The name of the animal constitutes the first attribute. There are 15 boolean features corresponding to the presence of hair, feathers, eggs, milk, backbone, fins, tail; and whether airborne, aquatic, predator, toothed, breathes, venomous, domestic, catsize. The numeric attribute corresponds to the number of legs.

Table 7 summarizes the clustering results. The Distinctness measure indicates better partitioning for the clusters generated by the b-coloring clustering approach (for the original partition as well as for the improved partition). This confirms that the utilization of dominating vertices finds more meaningful and well-separated clusters. In the other hand, the improved partition has the larger value. This indicates the pertinence of the greedy algorithm to improve the original b-coloring partition.

| method | # Clusters | Distinctness | $Dunn_G$ |
|---|---|---|---|
| re-coloring based | 7 | *0.652* | *1.120* |
| original b-coloring | 7 | 0.612 | 1.071 |
| agglomerative single-link | 2 | 0.506 | 0.852 |
| Hansen | 4 | 0.547 | 1.028 |

Table 7. the result of Zoo dataset.

### 5.2 Mushroom dataset

Each data record contains information that describes the 21 physical properties (e.g., color, odor, size, shape) of a single mushroom. A record also contains a *poisonous* or *edible* label for the mushroom. All attributes are categorical; for instance, the values that the size attribute takes are narrow and broad, while the values of shape can be bell, at, conical or convex, and odor is one of spicy, almond, foul, fishy, pungent etc. The mushroom database has many data items (the number of data items is 8124).The number of *edible* and *poisonous* mushrooms in the data set is 4208 and 3916, respectively. There are 23 species of mushrooms in this data set. Each species is then identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one.
Table 8 summarizes the results of the clustering obtained, over the mushroom data using the different clustering approaches.

| method | # Clusters | Distinctness | $Dunn_G$ |
|---|---|---|---|
| re-coloring based | 17 | *0.728* | *0.995* |
| original b-coloring | 17 | 0.713 | 0.891 |
| agglomerative single-link | 20 | 0.615 | 0.866 |
| Hansen | 19 | 0.677 | 0.911 |

Table 8. he result of Mushroom dataset.

Furthermore, we also analyzed the assigned objects in the clusters. Table 9 and Table 10 show the membership differences among the clusters by the previous b-coloring approach and the proposed approach in this paper. The clusters with bold italic characters represent the so-called *non-pure* clusters. These clusters are called non-pure, since they contain both poisonous and edible data items (mushrooms), and fail to separate them solely based on their features.

| Cluster ID | # of Edible | # of Poisonous | Cluster ID | # of Edible | # of Poisonous |
|---|---|---|---|---|---|
| 1 | 0 | 36 | 11 | 139 | 0 |
| ***2*** | ***96*** | ***464*** | 12 | 18 | 0 |
| ***3*** | ***695*** | ***72*** | 13 | 0 | 1296 |
| 4 | 768 | 0 | 14 | 224 | 0 |
| 5 | 1510 | 0 | 15 | 0 | 1728 |
| 6 | 220 | 0 | ***16*** | ***48*** | ***32*** |
| 7 | 145 | 0 | 17 | 192 | 0 |
| 8 | 0 | 288 | | | |
| 9 | 144 | 0 | | | |
| 10 | 9 | 0 | | | |

Table 9. details of cluster assignment for Mushroom dataset by the original approach

From these tables, we observe that almost all the clusters generated by both approaches are pure, except for the three clusters (Cluster 2, 3 and 16). This result also confirms that the utilization of dominating vertex contributes to generating to more meaningful and well-separated clusters.

| Cluster ID | # of Edible | # of Poisonous | Cluster ID | # of Edible | # of Poisonous |
|---|---|---|---|---|---|
| 1 | 0 | 36 | 11 | 107 | 0 |
| ***2*** | ***96*** | ***464*** | 12 | 16 | 0 |
| ***3*** | ***475*** | ***72*** | 13 | 0 | 1296 |
| 4 | 768 | 0 | 14 | 288 | 0 |
| 5 | 1728 | 0 | 15 | 0 | 1728 |
| 6 | 192 | 0 | ***16*** | ***48*** | ***32*** |
| 7 | 145 | 0 | 17 | 192 | 0 |
| 8 | 0 | 288 | | | |
| 9 | 144 | 0 | | | |
| 10 | 9 | 0 | | | |

Table 10. details of cluster assignment for Mushroom dataset by the proposed approach

## 6. Discussions

In our current approach we employ a greedy strategy tackle the re-coloring problem defined in Section 4.1. The major reasons for utilizing a greedy strategy is, as in other many approaches based on some greedy algorithms, we believe that it is useful as well as crucial for handling real world data, especially for large scale data. Based on this hypothesis, both the selection of vertex to be re-colored and the selection of the color to be assigned, is conducted in greedy fashion.

The other side of our greedy algorithm is that, besides it tries to improve the quality of partition while satisfying the constraints, there can still be better solutions for the re-coloring problem. If finding out better solutions is the most important (and, the only) interest, then, it would be possible to seek other much more expensive approaches. For instance, it might be possible to incorporate some kind of back-tracking for the re-coloring of the vertices. Such a recursive approach might be useful, both for the conceptual simplicity of the algorithm as well as the quality of the obtained solutions, in compensation for the incurred computational complexity.

In addition, there are many interesting issues to pursue:

1. more experiments and comparison for our algorithm on real world datasets, and
2. extension of our re-coloring approach for the critical vertices

As for (1), medical datasets or large scale image datasets seem interesting. As for (2), relaxing the constraints on the critical vertices seems promising for finding out better partition.

## 7. Conclusions

This paper has proposed a new greedy algorithm to improve the quality of clustering, while satisfying the constraints in the b-coloring of a specified graph. The previous b-coloring based clustering approach enables to build a fine partition of the data set (classical or symbolic) into clusters even when the number of clusters is not pre-defined. However, since it does not consider the quality of the clusters, besides obtaining the clusters in terms of the b-coloring of the graph, it was difficult to obtain better clusters explicitly. The proposed algorithm in this paper can complement this weakness. It conducts the re-coloring of the vertices (which correspond to data items) to improve the quality of the clusters, while satisfying the constraints. A greedy strategy is employed in the re-coloring process, both for the selection of vertex to be re-colored and the selection of the color to be assigned. We believe that utilization of a greedy strategy is useful as well as crucial for handling real world data, especially for large scale data.

The proposed greedy algorithm was tested over benchmark datasets from the UCI repository. The detailed results of the evaluations are reported and discussed. Through this evaluation, the effectiveness of the proposed greedy algorithm is confirmed. Especially, the results of experiments indicate that our approach is useful to offers a compromise between the inter-cluster separation and the intra-cluster cohesion.

## 8. Acknowledgments

The first author was supported by Canon Foundation in Europe Research Fellowship for his stay in France. The second author was supported by JSPS, Japan (PE 07555) for his stay in Japan. The authors are grateful to these grants. This work is partially supported by the grant-in-aid for scientific research (No. 20500123) funded by MEXT, Japan.

## 9. References

Bezdek, J.C. & Pal, N.R. (1998). Some new indexes of cluster validity. *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 28, No.3, 1998, pp.301-315

Elghazel, H.; Deslandres, V., Hacid, M.S., Dussauchoy, A. & Kheddouci, H. (2006). A new clustering approach for symbolic data and its validation: Application to the healthcare data. *Proceedings of ISMIS2006*, pp.473–482, Springer Verlag

Elghazel, H.; Yoshida, T., Deslandres, V., Hacid, M.S. & Dussauchoy, A. (2007). A new Greedy Algorithm for improving b-Coloirng Clustering. *Proceedings of GbR2007*, pp.228-239, Springer Verlag

Guenoche, A.; Hansen, P. & Jaumard, B. (1991). Efficient algorithms for divisive hierarchical clustering with the diameter criterion. *Journal of Classification*, Vol.8, pp.5-30

Guha, S.; Rastogi, R. & Shim, K. (1998). Cure: An efficient clustering algorithm for large databases. *Proceedings of the ACM SIGMOD Conference*, pp.73-84

Hansen, P. & Delattre, M. (1978). Complete-link cluster analysis by graph coloring. *Journal of the American Statistical Association*, Vol.73, pp.397-403

Hartigan, J. & Wong, M. (1979). Algorithm as136: A k-means clustering algorithm. *Journal of Applied Statistics*, Vol.28, pp.100-108

Blake, C.L. & Merz, C.J. (1998). UCI repository of machine learning database. University of California, Irvine. http://www.ics.uci.edu/~mlearn/MLRepository.html

Irving, W. & Manlov, D. F. (1999). The b-chromatic number of a graph. *Discrete Applied Mathematics*, Vol.91, pp.127-141

Jain, A.K.; Murty, M.N. & Flynn, P.J. (1999). Data clustering: A review. *ACM Computing Surveys*, Vol.31, pp.264-323

Kalyani, M. & Sushmita, M. (2003). Clustering and its validation in a symbolic framework. *Pattern Recognition Letters*, Vol.24, No.14, pp.2367-2376

Ng, R. & and Han, J. (2002). Clarans: a method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, Vol.14, No.5, pp.1003-1016

Zahn, C.T. (1971). Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, Vol.20, pp.68-86