# A Hybrid Heuristic Approach for Attribute-oriented Mining

Maybin K. Muyeba[1] , Keeley Crockett[1], Wenjia Wang[2], John A. Keane[3]
[1] *SCMDT, Manchester Metropolitan University, UK*
[2] *School of Computing Sciences, University of East Anglia, UK*
[3] *School of Computer Science, University of Manchester, UK*
{m.muyeba, k.crockett}@mmu.ac.uk, Wenjia.Wang@uea.ac.uk, jak@cs.man.ac.uk

**Abstract.** We present a hybrid heuristic algorithm, *clusterAOI,* that generates a more interesting generalised table than obtained via attribute-oriented induction (AOI). AOI tends to overgeneralise as it uses a fixed global static threshold to cluster and generalize attributes irrespective of their features, and does not evaluate intermediate interestingness. In contrast, clusterAOI uses attribute features to dynamically recalculate new attribute thresholds and applies heuristics to evaluate cluster quality and intermediate interestingness. Experimental results show improved interestingness, better output pattern distribution and expressiveness, and improved runtime.

***Keywords:*** *Induction, heuristic, threshold, interestingness, cluster, algorithm.*

## 1 Introduction

*Pattern interestingness* is determined by an objective measure [18] or by subjective user interpretation [15]. Threshold-driven algorithms [18, 12] generate many rules which need to be filtered to determine interestingness [15]. Attribute-oriented induction (AOI) [9] extracts high-level generalised rules by repeatedly replacing and clustering [14, 16] attribute values using domain knowledge[†] [9, 17]. AOI uses attribute and relation generalisation thresholds to limit the number of distinct attributes and rules generated.

Problem and approach: we aim to obtain generalized and hence more interesting rules. AOI overgeneralises to "ANY" values [3, 13, 14] as it uses a

[†] Use of domain knowledge has been limited [20] and is acknowledged as a hard problem [5].

fixed global static threshold to generalize attributes irrespective of their features and does not dynamically evaluate interestingness. Hence, the key idea here is to consider attribute features, to dynamically recalculate new thresholds, and to apply heuristics to evaluate cluster quality and intermediate interestingness. The most interesting rules consist of mostly interior concepts [6, 7, 14].

This paper presents *clusterAOI,* a hybrid heuristic algorithm based on [16], which produces a more interesting generalised table than AOI. A three-fold strategy is used: (1) generalise conservatively [14] selected clusters of attribute values that share common properties and satisfy a newly computed local attribute threshold; (2) evaluate intermediate interestingness result for each attribute and of the algorithm, using heuristic functions [16]; (3) apply Kullback-Leibler (KL) divergence to the output [10]. Experiments show improved interestingness (up to 4 times), better output pattern distribution and expressiveness (about 1.5 times), and improved runtime (about 2 times).

The approach is as follows: (1) *pre-cluster AOI* analyses attribute features to dynamically generate local thresholds; (2) *intermediate-clusterAOI* uses probabilistic semantic similarity between clusters of attribute values and evaluates cluster interestingness, resulting in improved interestingness and runtime; (3) in *post-clusterAOI* the final output table's interestingness is determined using an interestingness heuristic (global harmonic mean) and KL.

As an example we apply AOI and clusterAOI to a breast cancer dataset [19] (Table 1). We calculate KL for divergence and cluster quality (CQ). clusterAOI gave 0% overgeneralisation while AOI gave 50%; KL was 1.7 times higher and CQ 3 times higher. clusterAOI also produces twice as many informative rules (NOT-ANY), i.e. 100% compared to 50% for AOI. Similar weaknesses were highlighted in [21]. Overall, clusterAOI improves pattern understandability, intelligent interpretation and interestingness..

**Table 1**. *Comparing final output on non-ANY values, breast cancer dataset [19]*

| **Algorithm** | **cellSize** | **bNuclei** | **nNuclei** | **Mitoses** | **Count** | **%ANY** | **%not-ANY** |
|---|---|---|---|---|---|---|---|
| **AOI** | aboutAve | AboutAve | Any | Any | 485 | 50 | 50 |
| | aboutAve | AboveAve | Any | Any | 93 | | |
| **G.Thr=2, KL=0.63, CQ=11.59** | | | | | | | |
| **clusterAOI** | aboutAve | AboutAve | AboutAve | AboutAve | 483 | 0 | 100 |
| | aboutAve | AboveAve | AboutAve | AboutAve | 99 | | |
| | AboveAve | AboveAve | AboveAve | AboutAve | 71 | | |
| **G.Thr=2, KL=1.08, CQ=34.6** | | | | | | | |

The rest of the paper is structured as follows: related work is discussed in Section 2; Section 3 presents prerequisites and definitions; Section 4 introduces pre-clusterAOI; Section 5 presents intermediate and post-clusterAOI; Section 6 describes experimentation; and conclusions are given in Section 7. A running example of table 2 is extended as each aspect of the approach is discussed.

## 2 Related Work

AOI algorithms [3, 4, 8, 9, 13, 14] stop generalisation when thresholds (the interestingness measure) are reached, and do not consider attribute features and proprieties [3][18]. For pre-AOI, [21] removes discriminating data that may affect interestingness. Others [1][7] analyse depths and weighted heights of concept hierarchies to determine interestingness, but only use a single fixed weight value for interior concepts which may vary between hierarchies. For intermediate-AOI, [3] uses multiple-level support thresholds per attribute and order generalised tuples according to association strength. Others [1][11] select the next attribute generalisation path to follow but are computationally intensive. Repeating attribute values are preserved in [13], producing many output rules. In post-AOI [6, 16], the number of interior concepts in the output is used to evaluate interestingness using only the original global thresholds.

## 3 Prerequisites and Definitions

KL is an information divergence measure between two probability distributions (uniform and actual): higher values show good distribution and variety of output values, indicating improved interestingness [10]. Given m tuples $T = \{t_1,..,t_m\}$ and actual probabilities $\{p_1,..,p_m\}$, the divergence is $KL(T) = \log_2 m - \sum_{i=1}^{m} p_i \log_2 p_i$, where $KL(T) \geq 0$, bounded by $\log_2 m$.

We apply an interestingness heuristic function CQ to the top $k$ rules of the output (Equation 7, section 5.2), as in [1, 7][‡]. clusterAOI addresses interestingness as follows: let relation $R$ be defined on dataset $D \subseteq R$ with

[‡] Notation is collected in Appendix E.

$n$ tuples; attribute $A_i$ and attribute hierarchy $H_i$ pairs exist for m attributes i.e. $\{(A_1, H_1), (A_2, H_2), .., (A_m, H_m)\}$, $A_{m+1}$ is a count of tuples in $R$ and $t$ is a global threshold. Then $\sum | A_{m+1}, \varnothing | = n,$ with domain values $Dom(A_{m+1}, \varnothing) \in Z^+$, with $H_{m+1} = \varnothing$. Given a generalisation space $B_i = A_i \cup H_i$ for each attribute, we use entropy function $\nabla(A_i)$ and entropy values to generate new local thresholds $\{L.thr_i : L.Thr_i \geq G.Thr, i = 1, .., m\}$, for each $A_i$, where $G.Thr$ is the global threshold. $L.Thr_i \geq G.Thr$ means that each attribute should have at most $| L.Thr_i |$ distinct values, ensuring no overgeneralisation takes place. With a description language $\mathrm{L} = (\mathrm{B_i}, f)$, there is a level-by-level "nearest parent" generalisation function $f : B_i \to Dom(H_i)$ and a partial order $(\prec, B_i)$ for finding descriptions $\{\varphi_1, \varphi_2, .., \varphi_k\}$ in $B_i$. For the parent of a cluster $\{\varphi_1, .., \varphi_j\}$, $\varphi' = min\ \{ f(\varphi_1), .., f(\varphi_j) \}$ is a new description (nearest parent) for both $\varphi_i$ and $\varphi_j$, leading to Definition 1.

**Definition 1**. *Generalisable cluster*. Given a cluster $C_j = \{c_1, .., c_n\}$ of values for attribute $A_i$ and local threshold $\alpha$, cluster $C_j$ is *generalisable* if $| C_j | \geq \alpha$ and $f(c_k) = f(c_l)$, $\forall k, l \leq n, k \neq l$.

Generalisation of each attribute stops when its optimal value (*a local interestingness value,)* is reached (Definition 2, section 5.1), and in the global case when a global optimal value is encountered (Theorem 1, section 5.1). These values are derived by applying heuristic functions to attribute clusters. Without loss of generality, interestingness [16] can be described by both distance and cluster tightness [17] depending on tuple distribution in a summary table [10] (a cluster of attribute values). The agglomerative hierarchical clustering distance $\delta_n$ and tightness $\tau_n$ functions [17] are used for overall interestingness: $G_n : (\delta_n, \tau_n) \rightarrow \Re^+$. These functions exhibit both monotone and anti-monotone properties during generalisation [17]. Therefore, the problem of mining generalised patterns is a 4-tuple $(\nabla, I^i{}_L, f, I^T{}_g)$ (See appendix E) defined as follows:

(1) Find attribute significance using a distance linear function $\nabla(A_i)$ and generate new local threshold $L.Thr_i$; discussed in Sections 4.1 and 4.2;

(2) Find local attribute interestingness in iteration $k$ and aggregate values using a cluster tightness function $I^i{}_L(A_i)$ discussed in Sections 5.1;

(3) Generalise values using distance function $f(B_i)$ subject to $L.Thr_i$;

(4) Find global interestingness of $T$ by aggregating local interestingness values from (2) using function $I^T{}_g(..)$; discussed in Sections 5.2-5.4.

## 4. Pre-clusterAOI

Pre-clusterAOI aims to find each attribute's local threshold L.Thr (from G.Thr) using attribute features such as concept hierarchy and distinct values.

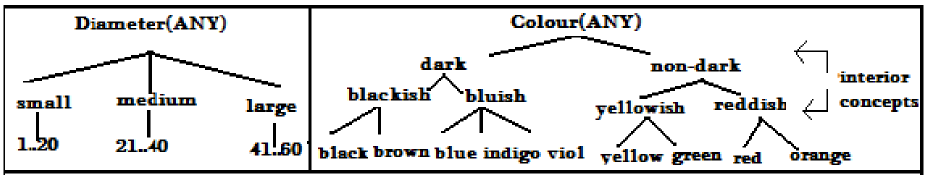


**Figure 1** Concept hierarchies for Diameter and Colour

**Table 2.** *Ball data*

| *Diameter* | *Colour* |
|---|---|
| *2* | *Red* |
| *7* | *Blue* |
| *34* | *Yellow* |
| *25* | *Green* |
| *28* | *Orange* |
| *8* | *Violet* |
| *16* | *Red* |

**Table 3.** *Prime table*

| *Diameter* | *Colour* | *Count* |
|---|---|---|
| *Small* | *Reddish* | *1* |
| *Small* | *Bluish* | *1* |
| *Medium* | *Yellowish* | *1* |
| *Medium* | *Yellowish* | *1* |
| *Medium* | *Reddish* | *1* |
| *Small* | *Bluish* | *1* |
| *Small* | *Reddish* | *1* |

*Example (1): The concept hierarchy, for a soccer dataset with 2 attributes, is shown in Figure 1 with interior concepts as non-leaf or root nodes. Table 2 shows that Diameter and Colour have 7 and 6 distinct values respectively, while prime table 3(the first generalised table) and table 4 have 2 and 3 distinct values respectively(See section 5.2 for generalisation process).*

### 4.1 Attribute feature significance

We adapt a query-based approach [14] to determine attribute significance from distinct values in $D$ and concept hierarchy node values in $H_i$ [6, 16]. Attribute significance is used to compute local thresholds by a tightness function $\tau_1(C_i, A) = \nabla(A)$; entropy $e_i(A_i)$ is calculated by Equation 1:

$$\tau_1(c_i, A) = \nabla(A_i) = -(\sum_i^m \log p(c_i) + \log \frac{|D|}{dist(D, A_i)}) = e_i = sig(A_i) \qquad \text{Eq (1)}$$

where $dist(D, A_i)$ or $dist(A_i)$ is the distinct values of attribute $A_i$ in D, taking the absolute value of Equation 1 and each $c_i \in H_i$ is a non-leaf concept. Note that $dist(D, A_i)$ is initially calculated from table 2. Merging table 3 and generalising "Colour" further gives table 4, finally merged to give table 5.

<u>Example (2):</u> *We calculate attribute significance* $sig(A_i)$ *values as follows:*

$$e_{diamter} = sig(diameter) = \nabla(A_{diamter}) = 4.76$$

*Similarly,* $e_{colour} = 9.78$, *meaning that Colour is more significant than Diameter. From this, new thresholds can be calculated.*

***Table 4.*** *Merged prime table*

| ***Diameter*** | ***Colour*** | ***Count*** |
|---|---|---|
| *Small* | *Bluish* | *1* |
| *Small* | *Bluish* | *1* |
| *Medium* | *Non-dark* | *1* |
| *Small* | *Non-dark* | *4* |

***Table 5.*** *Final table*

| ***Diameter*** | ***Colour*** | ***Count*** |
|---|---|---|
| *Small* | *Non-dark* | *4* |
| *Small* | *Bluish* | *2* |
| *Medium* | *Non-dark* | *1* |

## 4.2 Multi-threshold generation

Using equation 1 for entropy $e_i$, different local thresholds ($L.Thr_i$) can be generated for each attribute $A_i$ by a linear function $D_1(e_i, G.Thr)$ in $e_i$, where $E = MAX(e_1, e_2, .., e_m)$. Equation 2 states that $L.Thr_i \geq G.Thr$ for each attribute (i.e. in the worst case each is generalised using G.Thr). Hence, choosing the maximum entropy value is justified.

$$D_1(e_i, G.Thr) = \frac{(G.Thr * E) + |E - e_i|}{E} = G.Thr + \frac{|E - e_i|}{E} = L.Thr_i \qquad \text{Eq (2)}$$

<u>Example (3):</u> *Using Table 2 and* $G.Thr = 2$*, we obtain entropy values for Diameter* $e_1 = 4.76$ *and Colour* $e_2 = 9.78$ *with* $L.Thr_i$ *as follows:*

$$L.Thr_{colour} = \frac{(2*9.78) + |9.78 - 9.78|}{9.78} = G.Thr = 2$$

$$L.Thr_{diameter} = \frac{(2*9.78) + |9.78 - 4.76|}{9.78} = 2.51 > G.Thr$$

*where* $Max(4.76, 9.78) = 9.78$. *With value rounding,* $L.Thr_{colour} = 2$ *as* $|E - e_i| = 0$ *while* $L.Thr_{diameter} = 3$. *Generating* $L.Thr_i$ *affects each attribute's interestingness (shown in Section 5).*

## 5 Intermediate and post-clusterAOI

This section considers interestingness evaluation during generalisation (intermediate-clusterAOI) and of the final output table (post-clusterAOI).

### 5.1 Attribute Interestingness

Higher level concepts are more interesting than leaf nodes [6]; in contrast, "*ANY*" is uninteresting [7]. This represents a bounded interestingness problem (Proposition 1). AOI [9] uses a distance-driven generalisation method and interestingness should naturally be a distance function [16]. For simplicity, all values in the space $B_i$ consisting of attribute and concept hierarchy values are referred to as *concepts* unless specific meanings are required.

***Proposition 1. Concept value interestingness:*** A concept value $y \in B_i$ is *interesting* if it is an interior concept. See Appendix A for proof.

Example (4): *Figure 1 (Section 4) shows a concept hierarchy for Diameter with interior concepts {small, medium, large}* $\subset \{ANY\}$. *Novices may find range [1..60] and* $"ANY"$ *values both meaningless and uninteresting. However, they become interesting when diameter size can be determined without knowing specific measurements of the ball. Definition 1 enables such determinations to be made.*

Given Proposition 1, generalised attribute $A_i$ has concept collections $\bigcup_{k=1}^{n} c_k$ that give the highest interestingness value, an aggregation of individual interestingness values. Section 5.2 defines further heuristics for intra-cluster tightness $I_1^k$, inter-cluster similarity $I_2^k$ and cluster quality $I_3^k$ to calculate highest aggregated interestingness at any $k$ th iteration [16], called local (L) interestingness, for each $A_i$, denoted $[I_L^i(A_i)]^k$ or simply $l_i^k(X^k)$. When L is found, generalisation stops. As values $X^k$ vary between attributes at each iteration, we aggregate these values using a simple harmonic function.

**Definition 2**. *Local Attribute Interestingness*. Given an attribute $A_i$, local heuristic values $X^k = \{x_1^k, x_2^k, x_3^k\}$, and a harmonic aggregation function $l_i^k(X^k)$ with value $v^k = l_i^k(X^k)$ at the $k$ th iteration, the local attribute interestingness of the attribute is $v^{k+1} = l_i^{k+1}(X^{k+1})$ obtained at the $(k+1)$ th iteration, where $v^k = \max\{v^0,..,v^k,v^{k+1}\}$ and $v^k \geq v^{k+1}$.

Note from Definition 2 that $v^k$, the maximum local interestingness (optimum) value, is greater than $v^{k+1}$. At this point, no further generalisation occurs.

***Lemma 1***. *Interestingness function $I$ monotonically increases for the first $k$ iterations to an optimal value $v^k \in \Re^+$ and then monotonically decreases from the $k+1$ th iteration*. See Appendix B for proof.

It is easy to determine global interestingness simply by a harmonic aggregation at each iteration and finding the maximum so far.

<u>Example (5)</u>: *From Table 2, G.Thr=2, local and global interestingness values at each iteration were: Diameter {0.0, 0.14, 0.66, 1.0,..}, Colour {0.0, 0.29, 1.5, **0.78**} and clusterAOI {0.0, 0.21, 1.1, **0.91**} respectively. Diameter has no optimal value yet but Colour converged at iteration 3 at 0.78 (optimal value 1.5) and clusterAOI at 0.91 (optimal value 1.1). Although Diameter monotonically increases, further generalisation stops at iteration 3 at 1.0. This prevents overgeneralisation of both attributes (local case) and clusterAOI*

*(global case). Further, run-time performance is improved. Lemma 1 helps to formulate a theorem based on the local interestingness criteria.*

***Theorem 1.*** *(Global Interestingness).* Given attributes $\{A_i\}, 1 \le i \le m$ of a table T, each attribute has a local interestingness value $v_i^k$ for some iteration $k > 0$, global interestingness $v_g^{k+1}$ is the $(k+1)$ th value computed when the maximum value of aggregated local interestingness values in the $k$ th iteration has been computed. The aggregated maximum value in $v_g^k = \max\{\bigcup I_g^T(v_1^k, v_2^k, .., v_m^k)\}$ and global interestingness is reached when $v_g^k \ge v_g^{k+1}$ (Equation 7, section 5.2). See Appendix C for proof.

To reiterate, we calculate attribute interestingness (Equation 3, [16]) of a value c by a distance-based linear metric, $I_c : [0,1] \rightarrow [0,1]$. For example, a leaf concept $x_i \in A_i$ for attribute $A_i$ has interestingness value $Ix_i = 1 - \frac{d(x_i, "ANY")}{depth("ANY")} = 0$. Similarly, $I_{c=ANY} = 1 - 0 = 1$, a complement of the leaf concept mapping. This function is unreliable in determining interestingness as $"ANY"$ has the maximum value 1 by the mapping function.

$$I_c = 1 - \frac{d(c, "ANY")}{depth("ANY")}, c \ne "ANY" \qquad \text{Eq (3)}$$

Equation 3 satisfies the following three axioms for any given universal set X:

***Axiom 1:*** *Boundary conditions*: $I_c(\mu_\varnothing = 0) = 1$ and $I_c(\mu_x = 1) = 0$.

***Axiom 2:*** *Monoticity*: $I_c(a) > I_c(b)$ if $a < b$. For example, given two values $x_1 = "red" \prec x_2 = "ANY"$, $a = \mu_{x1} = 0, b = \mu_{x2} = 1$, then $I_{red}(0) = 1 > I_{ANY}(1) = 0$. This metric is unreliable for interestingness: $\exists c$ such that $"red" \prec c \prec "ANY"$, and if $\mu_c = 0.5$ then $I_c(0.5) > I_{c=ANY}(1)$ and $I_{c=red}(0) > I_c(0.5)$. This *violates* Proposition 1 and Lemma 1. A non-linear metric (see Equation 4) may be more suitable.

***Axiom 3:*** *Involutive*: It is easily shown that $I_c(I_c(0)) = 0$ or $I_c(I_c(1)) = 1$

We infer that interestingness is non-linearly distributed between leaf and root nodes. A probability density function, with input from Equation 3, determines interestingness of a value, and is given as Equation 4:

$$H(I_c) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x_i-\mu)^2 / (2\sigma^2)} \qquad \text{Eq (4)}$$

where $0 \le I_c \le 1$ with mean and variance values $(\mu, \sigma)$, $\forall c \in B_i$ .

*<u>Example (6)</u>: Using Table 1, mean $\mu = 0.524$, variance $\sigma = 0.069$ for Colour and $\mu = 0.625$, $\sigma = 0.0625$ for Diameter. Observe that Equation 4 gives $H(I_{red}) = H(0) = 1.48 * 10^{-12}$ while $H(I_{ANY}) = H(1) = 0.022 * 10^{-12}$. So interestingness of interior node "reddish", $I_{reddish} = 0.33$, $H(I_{reddish}) = 0.124$, and any cluster $C$ is given by the value $\sum_{i=1}^{n} H(c_i)$. Note $H(I_{reddish}) \ge H(I_{red}) \ge H(I_{ANY})$, thus confirming that root and leaf nodes*

*are less interesting than interior nodes. clusterOAI therefore aims to generalise values to interior concepts.*

## 5.2 Generalisation Process

For each attribute we find and generalize clusters of values that share common parents [16] and then computing interestingness heuristics (L2.7-L2.10, Figure 2); see Tables 3-6. Generalised clusters are stored in a hash table. As clusterAOI may not generalise all clusters, another approach is necessary (Section 5.4) based on Axiom 2. To derive interesting patterns we measure cluster tightness (Equation 5); [17] presents a general theory of monotonically non-increasing tightness functions $\tau_n$ that measure the number of attribute values in clusters i.e. if $c \prec c'$, $\tau_n(c) \succ \tau_n(c')$. We require $\tau_n(c') \succ \tau_n(c)$ to hold for $c \prec c'$, where the tightness function increases monotonically to an optimal value. New clusters are generated by generalising common parent clusters, following a similar generalisation process as in AOI. The output stores new parent clusters and similarity, tightness and CQ heuristics (L2.9, Figure 2), aggregated using Equation 5. The final generalised table is produced in step 3, L3.6 of Figure 2. In Table 2, cluster {2, 7, 8, 16} is generalised to its nearest common parent "small" because $4 > L.Thr_{diameter} = 3$. This guarantees conservative generalisation, cluster by cluster. clusterAOI uses this step to

## 5.3 clusterAOI algorithm

Figure 2 shows clusterAOI. Step 1 performs pre-clusterAOI.

```
Input:    dataset D with attributes Ai; Concept hierarchies Hi, Global threshold t
Output:   A compact table T, Kullback measure KL, Cluster measure $I^T_g$
BEGIN
Step 1. Get Ai from D, Find attribute significance and new local thresholds iter←1
Step 2. Process attributes
L2.1    stillInteresting=true
L2.2    convergenceValues [Ai]= false
L2.3    While (stillInteresting) // check Global interestingness
L2.4      For each Ai in D
L2.5        attrConv= convergenceValues (Ai)
L2.6      if ( not(attrConv))
L2.7            D←processAttribute(D, Hi, Ai)
L2.8            D←mergeData(D)
L2.9            attrHeur[i]←computeHeuristics(interC,intraC, $I^i_L$ ) // [16]
L2.10           hashStore=IterationObject(attrHeur[i], iter)
L2.11            harm_total←0
L2.12           $I^i_g$←0
L2.13    For each Ai //check attribute convergence
L2.14        if(iter>0)
L2.15           attrConv←isAttribConv(iter, i, hashStore)
L2.16           if(attConv= true)// Check Local interestingness
L2.17             convergenceValues(i)←attrConv
L2.18      $I^i_g$ ← computeGlobal_I(hashStore (iter, Ai)) // eqn (5)
L2.19      harm_total← harm_total +$I^i_g$
L2.20     $I^D_g$ =harm_total/m  //eqn (6)
L2.21     setHarmValues($I^D_g$ ) )
L2.22    stillInteresting←isInteresting(getHarmValues())
L2.23    iter←iter+1
Step 3. Generate final table
L3.1     D←merge and Copy(D)
L3.2     D←generaliseLeaf(D)
L3.3     T←topKRules(D, newThreshold)
L3.4     KL←KullbackMeasure(T)
L3.5     $I^T_g$←TableClusterQuality(T) //eqn (7)
L3.6     Output(T, KL, $I^T_g$)
END
```

**Figure 2** The clusterAOI algorithm

produce better patterns than AOI. Let $C_{ij}$ be cluster $j$ for a group of values from attribute $A_i, 1 \le j \le n,$ for n tuples with clusters $C = \{\bigcup_{j=1}^{k} C_{ij}\}$ as a collection of all clusters for the attribute, with similar attribute properties. e.g.

equal distances from the root node and the same parent nodes. An attribute's global interesting value is obtained using cluster similarity (*IntraC*), tightness (*InterC*) and total local interestingness ($I^i{}_L$), using Equation 5, ($\mathrm{I} = (\mathrm{I}_1, \mathrm{I}_2, \mathrm{I}_3)$ respectively)§. After computing $I^i{}_g$, we store the global interestingness values $I^D{}_g$ for all table attributes (Equation 6). To further generalise an attribute, the latest and previous $I^i{}_g$ value are compared to determine the next step. Global interestingness values are compared in the same way. Comparing values this way avoids overgeneralisation. The cluster quality of a table T is a summation of all local $I^i{}_L$ values (Equation 7).

$$I^i{}_g(I) = \frac{n}{\sum_{i=1}^{n} 1/I_i} \qquad \text{Eq (5)}$$

$$I^D{}_g = \frac{\sum_{i=1}^{m} I^i{}_g}{m} \qquad \text{Eq (6)}$$

$$I^T{}_g = \sum_{i=1}^{m} I^i{}_L \qquad \text{Eq (7)}$$

Step 2 sets global interestingness, processes each attribute, merges resulting values and stores $I^i{}_L$ values (Equation 5). Each iteration makes parent

§ Defined as $I_1(C) = \frac{\alpha}{|\bigcup_{i=1}^{k} C_{ij}|}, \alpha = 1$, $I_2(C) = \sum_{\mathrm{i}=1}^{\mathrm{k}} |\mathrm{C}_{\mathrm{ij}}| - |\bigcup_{\mathrm{i}=1}^{\mathrm{k}} \mathrm{C}_{\mathrm{ij}}|$, $I_3(C) = \sum_{i=1}^{n} H(c_i)$

clusters per attribute, generalises and processes them per attribute (L2.7, Figure 2). *computeHeuristics()* (L2.9) computes interestingness heuristics. We check each attribute's local convergence (L2.15) and global convergence (L2.22) using the heuristics. Global interestingness is checked by the global harmonic mean of all local interestingness values (L2.20, Equation 6). Step 3 merges the table rows into a new smaller table.

***Table 6.*** *Clusters for Diameter and Colour at generalisation iteration 1*

| ***Diameter*** | ***IntraC*** | ***InterC*** | $I^i{}_L$ | $I^i{}_g$ ***(Eqn. 5)*** |
|---|---|---|---|---|
| *[2, 7, 8, 16] →small*<br>*[25,34, 28]→medium* | *0.143* | *0.0* | *0.0* | |
| | | | | |
| *TOTAL* | | | | *0.143* |
| ***Colour*** | | | | |
| *[red, orange, red]→reddish*<br>*[blue, violet]→bluish*<br>*[yellow, green]→yellowish* | *0.167* | *1.0* | *0.0* | |
| *TOTAL* | | | | *0.286* |
| | *Average H. Mean (eqn. 6)* | | | *0.429/2=0.215* |

<u>Example (7):</u> *Table 6 shows the initial clusters generated using a global threshold G.Thr=2. The set of global harmonic means are stored for the entire generalisation process as sets of vector values. The first iteration's global harmonic mean is 0.215. Local interestingness values for Diameter were* [0.0,0.143,0.66,1.03] *and for Colour* [0.0,0.286,1.5,0.78]. *Equation 5 gives the second local harmonic mean of Colour as* $\frac{2}{(1/0.167)+(1/1)} = 0.286$. *Local cluster interestingness values ($I^i{}_L$) are zero as we only have leaf concepts. Further, note that the algorithm only iterates to iteration 3 up to global harmonic value 0.907 in the list [0.0, 0.215, 1.08,* ***0.907****,..] and generalisation*

*stops. Colour essentially converges at iteration 3 (as 1.5>0.78) in the list*[0.0,0.286,1.5,0.78]. *As Diameter has not converged yet (as 0.66<1.33), more interestingness can be found after iteration 3. Consequently, as global algorithm convergence occurs at iteration 3, no further generalisation of Diameter occurs. This prevents overgeneralisation and further iterative steps, and saves execution time.*

The complexity analysis of clusterAOI is discussed in Section 6.4.

## 5.4 Post-clusterAOI Interestingness

*Example (8): Table 7 compares outputs: each row represents a rule in descending order of tuple numbers. clusterAOI recalculates thresholds and gives three rules containing interior concepts while AOI overgeneralises Diameter to "ANY" and gives two rules. clusterAOI is superior to AOI: global interestingness ($I^T{}_g$, 5th column) is 3 times and KL is 1.6 times better.*

***Table 7.*** *clusterAOI and AOI final table comparisons*

| | Diameter | Colour | Count | $I^T{}_g$ (Eqn. 7) | KL |
|---|---|---|---|---|---|
| clusterAOI | Medium | Non-Dark | 4 | 4.06 | 1.38 |
| | Small | Bluish | 2 | | |
| | Small | Non-Dark | 1 | | |
| AOI | ANY | Non-Dark | 5 | 1.34 | 0.86 |
| | ANY | Dark | 2 | | |

In clusterAOI, ungeneralisable attribute clusters may appear as leaf concepts in the output [16]. To improve overall interestingness, *generaliseLeaf*() (L3.2, Figure 2) searches for an optimal generalisation point for any given leaf concept. Following Axiom 2, there is an interior level $l$ (or group of interior parent concept values at this level) which is more interesting than those at levels $l-1$ and $l+1$. For performance, we deterministically find the most "interior" level of a concept hierarchy, the 'median', and generalise the leaf to this level (Proposition 1 and Figure 1). $(depthOfHierarchy)/2$ generalisation steps are used for an even number of levels; one more for an odd number.

## 6. Experimental Analysis

Experiments have been performed in terms of KL measure, interestingness and runtime, with as shown in Appendix F. Experiments were run 5 times to obtain average results on an Intel (R) Pentium (R) Dual 2GHz processor with 2GB RAM.

### 6.1 Census-income dataset (50K tuples, 3 attributes)

Table 8 compares performance with global thresholds from 1 to 10. Threshold 1 guarantees excessive generalisation in traditional AOI while higher thresholds do the opposite. For clusterAOI, mean and variance were calculated and their significance evaluated as follows: $sig(age) = 18.87, sig(educ) = 9.99$ and $sig(numWorked) = -1.63$, and local thresholds recalculated (Table 8, final column). Comparing global interestingness ($I^{T_g}$ values) in Table 8 (see also Figure 3(b)), clusterAOI is 2.65 times better on average. For thresholds 1 to 4, often a desirable level to set in AOI, clusterAOI interestingness is 12 times better, meaning it generates more interesting patterns. Higher thresholds (7 and over) appear to not differentiate interestingness between the two algorithms on this dataset. Unsurprisingly, threshold 1 gives interestingness of 22.36 and clusterAOI recalculated L.Thr as 2 and produced two rules with values (aged, average education, few) and (aged, basic education, few); in contrast, AOI produced one overgeneralised rule. The KL measure for clusterAOI is on average 1.76 times better for thresholds 1 to 4, and 0.91 times better for thresholds 1 to 10 (Figure 3(a)). The results indicate that clusterAOI generalises better for smaller thresholds (better distribution or divergence of output patterns) than for larger ones. clusterAOI is also 1.25 times faster (Figures 4(a), Table 8). Generally, for smaller thresholds, there is more clustering, generalisation and merging. Figure 4(b) shows how clusterAOI

**Table 8.** *clusterAOI and AOI results (50K Census-income dataset)*

| G.Thr | KL | | $I^T{}_g$ | | Runtime (x10sec) | |
|---|---|---|---|---|---|---|
| | *AOI* | *clusterAOI* | *AOI* | *clusterAOI* | *AOI* | *clusterAOI* |
| 1 | 0.00 | 0.97 | 0.00 | 22.36 | 26.20 | 20.50 |
| 2 | 0.86 | 1.54 | 5.80 | 10.22 | 26.20 | 20.50 |
| 3 | 1.46 | 1.94 | 0.002 | 13.19 | 26.20 | 21.00 |
| 4 | 1.46 | 2.24 | 0.018 | 23.10 | 26.20 | 21.00 |
| 5 | 2.26 | 1.83 | 0.0029 | 25.00 | 26.00 | 21.00 |
| 6 | 2.26 | 1.83 | 0.0029 | 25.60 | 26.00 | 21.00 |
| 7 | 2.26 | 1.83 | 26.00 | 25.60 | 26.00 | 21.00 |
| 8 | 2.94 | 1.83 | 26.00 | 25.60 | 26.00 | 21.00 |
| 9 | 2.94 | 1.83 | 26.00 | 25.60 | 26.00 | 21.00 |
| 10 | 2.94 | 1.83 | 26.00 | 25.60 | 26.00 | 21.00 |
| Avg. | # Times = 0.91 | | # Times = 2.65 | | # Times = 1.25 | |

iterates to a global optimal value $v_g^k = 4.5$ at iteration 3 before finally converging at iteration 4 at $v_g^{k+1} = 1.57$. Similarly, each attribute has a local optimal value $v^k$ before stopping at the next iteration (See Figures 5, 6, 7). Significance values were calculated as: age (18.87), education (9.98) and NumWorkedFor (-1.63) meaning that age has the smallest local threshold of the three attributes and should be generalised further. Note their convergence graphs in Figures 5, 6 and 7 are similar to the clusterAOI convergence pattern in Figure 4(b).

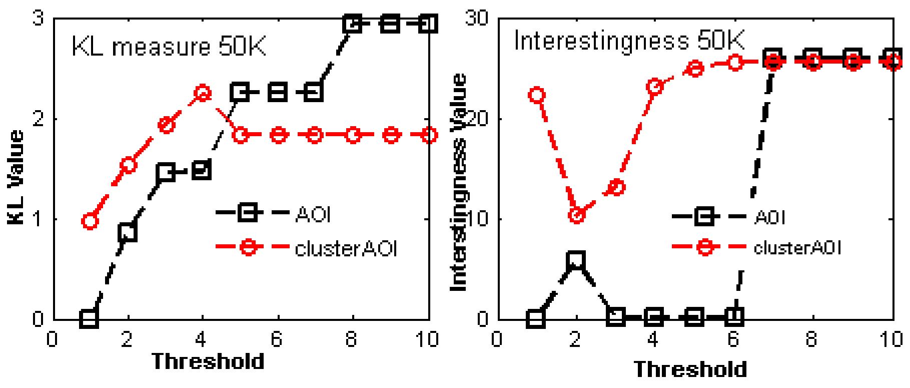


**Figure 3** 50K census-income: **(a)** KL **(b)** Global Interestingness

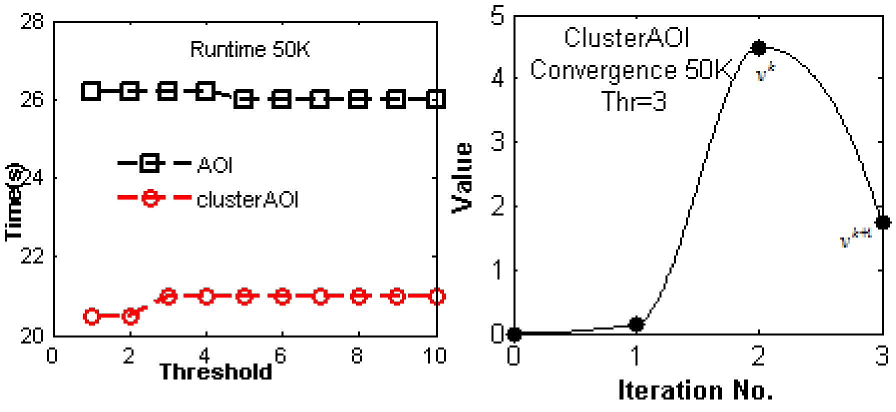


**Figure 4 (a)** Runtime 50K **(b)** Algorithm convergence

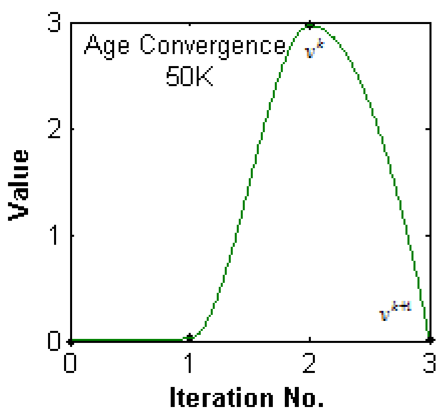


**Figure 5** Convergence of Age

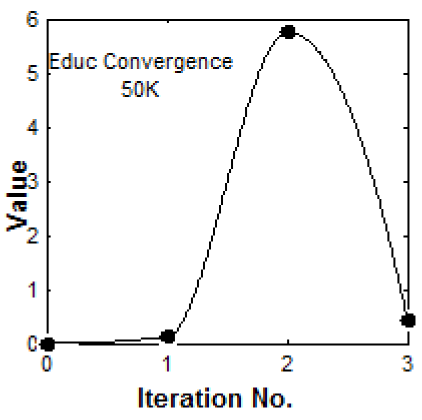


**Figure 6** Convergence of Education

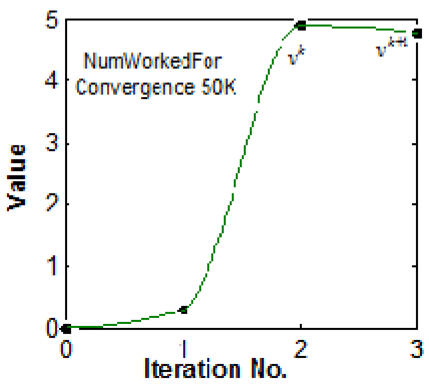


**Figure 7** Convergence for NumWorkedFor

## 6.2 Cancer-Wisconsin dataset (700 tuples, 4 attributes)

Section 1 discussed this dataset. As the dataset is small, values distribution (concept hierarchy and distinct attribute values) is narrow with a low value for KL. Each attribute has similar significance values other than mitoses: $sig(cellSize) = 5.04$, $sig(bareNuclei) = 5.04$, $sig(normalNuclei) = 5.04$ and $sig(mitoses) = 4.89$. As shown in Section 1, mitoses is not overgeneralised by clusterAOI even with the lowest significance, unlike by AOI. Convergence of the four attributes follows a similar pattern to Figures 5, 6 and 7. Table 9 shows clusterAOI is better than AOI in terms of global interestingness ($I^T{}_g$ values): 5.42 times better on average; KL for small thresholds (1 to 4): 1.5 times, overall it is 0.94 times better on average; and interestingness and runtime: 5.84 and 1.24 times respectively.

**Table 9.** *clusterAOI and AOI results (Breast cancer dataset [19])*

| **G.Thr** | **KL** | | $I^T{}_g$ | | **Runtime (x10sec)** | |
|---|---|---|---|---|---|---|
| | ***AOI*** | ***ClusterAOI*** | ***AOI*** | ***clusterAOI*** | ***AOI*** | ***clusterAOI*** |
| 1 | 0.0 | 0.66 | 0.0 | 23.2 | 18.4 | 16.40 |
| 2 | 0.63 | 1.08 | 17.4 | 34.6 | 17.0 | 16.00 |
| 3 | 1.08 | 1.36 | 23.0 | 46.4 | 17.0 | 16.00 |
| 4 | 1.08 | 1.36 | 0.75 | 46.4 | 19.0 | 16.00 |
| 5 | 1.44 | 1.36 | 0.90 | 46.0 | 19.0 | 16.00 |
| 6 | 1.59 | 1.36 | 1.06 | 46.4 | 20.0 | 15.60 |
| 7 | 1.71 | 1.36 | 1.21 | 46.3 | 21.0 | 15.60 |
| 8 | 1.83 | 1.36 | 1.36 | 46.4 | 20.0 | 16.00 |
| 9 | 1.92 | 1.36 | 1.51 | 46.4 | 21.0 | 16.00 |
| 10 | 2.01 | 1.36 | 31.88 | 46.4 | 27.0 | 16.00 |
| Avg. | # times = 0.94 | | # times = 5.42 | | # times = 1.24 | |

## 6.3 Census-income dataset (200K tuples, 6 attributes)

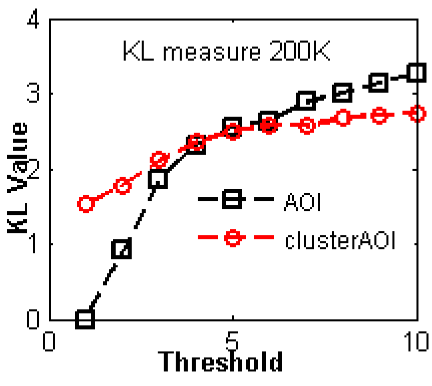
Significant values and distinct values ($dist(A_i)$) were calculated as follows: $sig(age) = 27.94$, dist(age)=91; $sig(education) = 8.01$, dist(education)=17; $sig(\det ailedHHold) = 64.20$, dist(detailedHHold)=38; major industrial code $sig(majIndCode) = 98.00$, dist(majIndCode)=24; $sig(instWeight) = -4.4$, dist(instWeight)=4 and $sig(numWorkedFor) = -3.6$, dist(numWorkedFor)=7. Global thresholds were set from 1 to 10 and performance results are shown in Table 10. clusterAOI generates patterns that are 4 times more interesting (Table 10, Column $I^T{}_g$ and Figure 9) and 1.04

times better in terms of pattern divergence (KL in Figure 8) than AOI. However, when averaging for thresholds 1 to 4 as previously, clusterAOI generates patterns that are 14 times more interesting; KL for thresholds 1 to 4 is about 1.5 times better than AOI. As data increases (unlike previously), KL (value divergence) for clusterAOI also increases. From Figure 8, the greater the divergence, the more interesting patterns are produced i.e. clusterOAI shows larger interestingness values (Figure 9). Hence, small thresholds may be used for many purposes e.g. readability, interpretability etc. clusterAOI is about twice as fast (Table 10, Figure 10); with lower thresholds (e.g. 1 to 4), and overall average run-time is about three times better. Note that for attribute

**Table 10.** *clusterAOI and AOI results (Census-income dataset 200K tuples)*

| G.Thr | KL | $I^T{}_g$ | | Runtime (x10sec) | | |
|---|---|---|---|---|---|---|
| | ***AOI*** | ***clusterAOI*** | ***AOI*** | ***clusterAOI*** | ***AOI*** | ***clusterAOI*** |
| 1 | 0.00 | 1.53 | 0.00 | 49.44 | 96.00 | 32.00 |
| 2 | 0.92 | 178 | 0.00 | 40.60 | 68.00 | 26.00 |
| 3 | 1.86 | 2.11 | 9.07 | 56.00 | 59.00 | 26.00 |
| 4 | 2.31 | 2.35 | 6.84 | 71.56 | 57.00 | 26.00 |
| 5 | 2.55 | 2.50 | 8.21 | 91.00 | 57.00 | 26.00 |
| 6 | 2.63 | 2.57 | 14.55 | 113.75 | 57.00 | 26.00 |
| 7 | 2.89 | 2.57 | 42.11 | 140.00 | 54.00 | 24.00 |
| 8 | 3.00 | 2.67 | 47.80 | 140.00 | 54.00 | 26.00 |
| 9 | 3.14 | 2.71 | 53.50 | 152.00 | 54000 | 26.00 |
| 10 | 3.27 | 2.74 | 59.20 | 165.80 | 54.00 | 26.00 |
| Avg. | # times = 1.04 | | # times = 4.20 | | # times = 2.31 | |

significant values lower than zero (e.g. instance weight), convergence $v^{k+1} = 1.98$ and optimal values $v^k = 1.99$ are close. This is similar to the experiment in Section 6.1 with attribute NumWorkedFor (Figure 7). Other positive significant values follow a nearly normal probability distribution (see Figure 6, Education attribute) similar to clusterAOI algorithm convergence at iteration 3 (Figure 11). Attribute convergence obtained similar patterns to those of Sections 6.1 and 6.2. Moreover, clusterAOI also follows a nearly



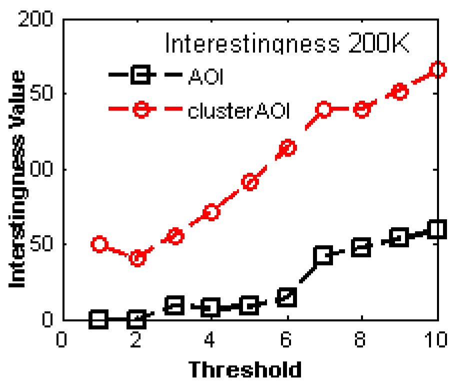
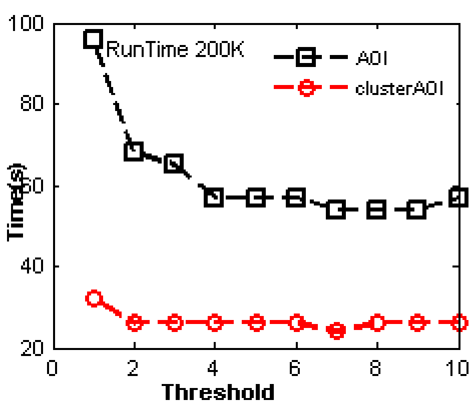


**Figure 8** KL: 200K **Figure 9** Global interestingness: 200K



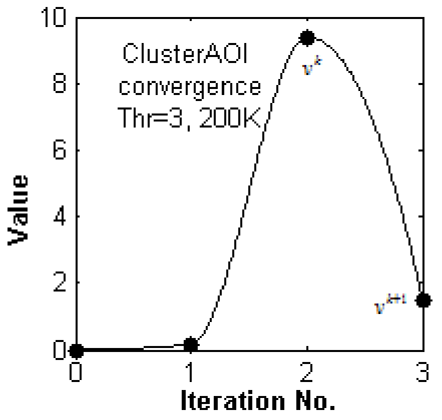


**Figure 10** Runtime: 200K **Figure 11** clusterAOI convergence:200K

normal probability distribution, showing the necessity of searching for optimal turning points (interestingness values) in the generalisation process (see Figures 4 (b) and 11). The same argument holds for positively significant attributes. clusterAOI and AOI are also compared in terms of NOT-ANY/ANY values. Using a global threshold of 3, clusterAOI on average obtained 100% NOT-ANY values (meaning 0%"*ANY*"values), across all datasets (see Table 11). In contrast, AOI gave 41% NOT-ANY values. Clearly clusterAOI derives more useful, meaningful and informative patterns than those obtained by AOI. These are consistent with Table 1 results.

**Table 11.** *Comparing NOT-ANY and ANY values (G.Thr=3)*

| **Dataset** | ***AOI*** | ***AOI*** | ***clusterAOI*** | ***clusterAOI*** |
|---|---|---|---|---|
| | ***ANY*** | ***not-ANY*** | ***ANY*** | ***not-ANY*** |
| Census-200K | 15 | 9 | 0 | 30 |
| Cancer-700 | 8 | 8 | 0 | 16 |
| Population 50K | 29 | 20 | 0 | 12 |
| **Avg % ANY** | **59%** | | **0%** | |
| **Avg % not-ANY** | **41%** | | **100%** | |

In summary, the experiments show that clusterAOI compared to AOI:

1. is superior in terms of runtime, interestingness and divergence (KL);
2. does not fluctuate between small and large datasets;
3. generates higher interestingness values, up to 13 times, with lower thresholds (G.Thr 1 to 5);
4. has a run-time complexity of approximately half (Section 6.4);
5. has better KL divergence, 1.5 times for smaller thresholds (up to 4)

## 6.4 Algorithm space and time complexity analysis

AOI has order complexities ranging from $O(np)$ [8], $O(n \log p)$[9] to $O(n)$ ([4][2]) for $n$ input tuples, $p$ generalised tuples and $p \leq \log n$.

**Lemma 2.** clusterAOI's time complexity is $O(np)$ with space complexity of $2 * O(n)$ or ( $O(np) + O(n)$ ), for $n$ input tuples, $p$ tuples in the prime table and child concept values stored in their parent cluster hash tables (See Appendix D for proof).

clusterAOI has better runtime than AOI because it scans the input data once and creates parent clusters during input (Figure 2: Steps 1, 2). Figures 4a and 10 show the differences in runtime performance.

## 7. Conclusions

A heuristic algorithm, clusterAOI, has been introduced that, when compared to AOI, improves expressiveness and interestingness, divergence and distribution of concepts in the output, and runtime. Experimental results show improvement on average of 1.0 to 1.57 times on KL, 4 times on interestingness and 2 times on runtime compared to AOI. The final output has better pattern distribution and is more expressive and meaningful than from AOI. clusterAOI uses pre-clusterAOI to determine attribute significance, intermediate-clusterAOI to preserve attribute interestingness and post-clusterAOI to evaluate output. clusterAOI has similar order complexity, $O(np)$, and storage requirements, $2*O(np)$ to established algorithms [2, 4]. Our approach is applicable to generalisation algorithms using concept hierarchies [12]. Further work will investigate heuristic optimisation to better exploit the search space.

## Acknowledgements

Thanks to Prof Howard J. Hamilton (University of Regina) for technical input.

## References

[1] B. Barber and H.J. Hamilton. A comparison of attribute selection strategies for attribute-oriented generalisation, Symposium on Methodologies for Intelligent Systems (ISMIS'97), pp 106-116 (1997)

[2] C.L. Carter and H.J. Hamilton. Efficient attribute-oriented generalisation for knowledge discovery from large databases, IEEE Transactions on Knowledge and Data Engineering, 10(2):193-208 (1998)

[3] Y.L. Chen, Y.Y. Wu and R. I. Chang. From data to global generalised knowledge, Decision Support Systems 52(2):295-307 (2012)

[4] D.W. Cheung, A.W. Fu and J. Han. Efficient rule-based attribute-oriented induction for data mining, Intelligent Information Systems 15:175-200 (2000)

[5] U.M. Fayyad, G. Piatetsky-Shapiro and R. Uthurusamy. Data Mining: the next 10 years, SIGKDD Explorations 5(2): 191-196 (2003)

[6] D.R. Fudger and H.J. Hamilton. A heuristic for evaluating databases for knowledge discovery with DBLEARN, Rough Sets and Knowledge Discovery, pp 29-39 (1993)

[7] H.J. Hamilton and D.R. Fudger. Estimating DBLEARN's potential for knowledge discovery in databases, Computational Intelligence, 11(2):280-296, (1995).

[8] J, Han. Towards efficient inductive mechanisms, Theoretical Computer Science, 133:361-385 (1994)

[9] J. Han. and Y. Fu. Exploration of the power of attribute-oriented induction in data mining, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, pp 399-421 (1996).

[10] R.J. Hilderman and H.J. Hamilton. Knowledge discovery and measures of interest. Kluwer Academic Publishers (2001).

[11] R.J. Hilderman, H.J. Hamilton and N. Cercone. Data mining in large databases using domain generalisation Graphs, Journal of Intelligent Information Systems, 13(3), 195-234 (1999)

[12] Y-F. Huang, C-M. Wu. Mining generalised association rules using pruning techniques,ICDM', pp 227-234 (2002).

[13] C-C. Hsu. Extending attribute-oriented induction algorithm for major values and numeric values, Expert Systems with Applications, 27(2):187-202, (2004).

[14] K. Julisch. Clustering intrusion detection alarms to support root cause analysis, ACM Transactions on Information and System Security, 6(4):443-471 (2003)

[15] B. Liu, K. Zhao, J. Benkler and W. Xiao: Rule interestingness analysis using OLAP operations, 12th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, USA, pp 297-306 (2006).

[16] M.K. Muyeba, K. Crockett and J.A. Keane. A hybrid interestingness heuristic approach for attribute-oriented mining, LNCS 6682:414-424 (2011).

[17] L. Pitt and R.E. Reinke. Criteria for polynomial-time (conceptual) clustering, Machine Learning, 2(4):371-396 (1988)

[18] P-N. Tan, V. Kumar. and J. Srivastava. Selecting the right objective measure for association analysis. Information Systems, 29(4): 293-313 (2004)

[19] UCI Datasets, http://archive.ics.uci.edu/ml/index.html accessed 27/11/12

[20] X. Wu. 10 Years of Data Mining Research: retrospect and prospect, IEEE International Conference on Data Mining (ICDM), pp 7, (2010).

[21] C. Yen-Liang and S. Ching-Cheng. Mining generalised knowledge from ordered data through attribute-oriented induction techniques, European Journal of Operational Research, 166(1), 221-245 (2005).

## Appendix A Proof of Proposition 1

**Proof**: A generalisation function is finite and bounded by values: a lower bound $a$ on the furthest leaf nodes and an upper bound $b$ on the root node (Figure 1). There are then at least one or more concepts $o_i \in H_i$ with property $a \prec o_i \prec b$ of some partial order. Following the work in [6] which states that leaf and root node values are uninteresting, it follows that for any interestingness function $I$ , $I(a) < I(o_i)$ and $I(o_i) > I(b)$ if $a$ is a leaf and $b$ is a root concept. Thus $o_i$ is an interesting interior concept. **{end proof}**

## Appendix B Proof of Lemma 1

**Proof:** The proof is a consequence of Proposition 1 and the example given. Assuming initial attribute interestingness as $\alpha \geq 0$, $\exists$ iteration $k$ where a cluster of concepts with heuristic values $X^k$ give a value greater than $\alpha$ i.e. $I_k^i(X^k) = v^k > \alpha, v^k \in \Re^+$. Given that further generalisation reduces the interestingness value of an attribute [16], a (k+1)th iteration, where $I_{k+1}^i(X^{k+1}) = b = v^{k+1} \leq v^k$. Thus interestingness $I$ increases in interval $[\alpha, v^k]$ and decreases in interval $[v^{k+1},..]$. {**end proof**}.

## Appendix C Proof of Theorem 1

***Proof***: Following Definition 2, let the aggregate harmonic values (local attribute interestingness values) be $\{v_1^k,..,v_m^k\}, 1 \leq i \leq m$ for $m$ attributes at iteration $k$. Generalisation interestingness functions [16] and heuristic aggregation functions are monotonically increasing [17] up to some optimal value at iteration k, denoted $v_g^k = I_g^T(v_1^k, v_2^k,.., v_m^k)$. If after further generalisation, iteration k+1 has value $v_g^{k+1} = I_g^T(v_1^{k+1},.., v_m^{k+1}) \geq v_g^{k+1}$, the decreasing function reaches a global interestingness value, else generalisation continues to iteration k+2 and so on **{end proof}.**

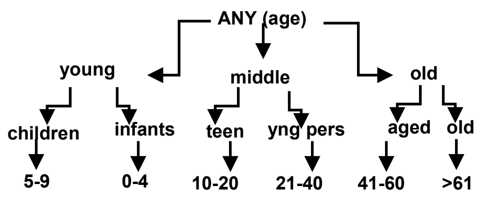
## Appendix D Proof of Lemma 2

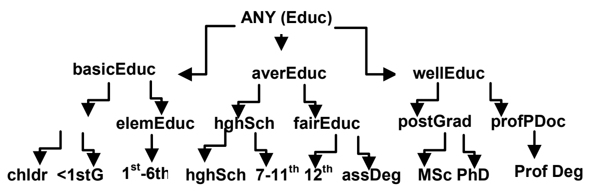
**Proof:** <u>*Time Complexity:*</u> Let $n$ be the input size, $p$ the tuples in a prime table and $m$ the number of attributes. AOI [9] reads the input into memory, generalises each attribute, sorts the table in $O(n \log n)$ time before merging the prime table. Thus, time complexity is $O(np) + O(n\log n) = O(n\log n)$, $p << n$. The total space is $O(n) + O(p) \approx O(n)$ and total time complexity $O(n\log n) \approx O(n\log n)$ $+O(p)$. At input, clusterAOI clusters attribute values $C_{ji}$ for each $A_i$, according to a nearest parent, on-the-fly and stores them under that parent in a hash table. Given a j$^{th}$ cluster as $C_{ji}$ for *k* parents, the time complexity is $O(k|C_{ji}|)$. For *h* clusters and global threshold G.Thr= $g > 0$, the time to store concepts is bound by $O(k * m * h * g * \sum | C_{ji} |$. Note that $m * \sum | Ci_j | = n$, $k$ <<n, $h << n$ and $g << n$ in the worst case where each C$_{ji}$ is generalised, *j=1,..,h*. Thus time complexity is $O(n)$. In addition, we generalise clusters of child concept values and insert tuples in a prime relation, $p << n$, giving order complexity $O(np)$. Further, to merge the prime relation by a merge sort and inserting in a final table of size *q*, complexity is *O (plogp)*O(q)* or simply $O(pq) \approx O(p)$*)*. Thus total order complexity is $O(np) + O(p)$. As *q <<p*, *q<< n*, and *p<<n*, the order complexity is at most *O(np)* in the worst case (and could be $O(n)$ in the best case). *Space Complexity:* Intuitively, we need *O(n)+O(n)* to store initial input and attribute child clusters (unless we only store child leaf value index positions as in [2]). The prime table size *O(p)* and final table *O(q)* are insignificant. Thus space complexity is $2 * O(np)$. **{End proof}**

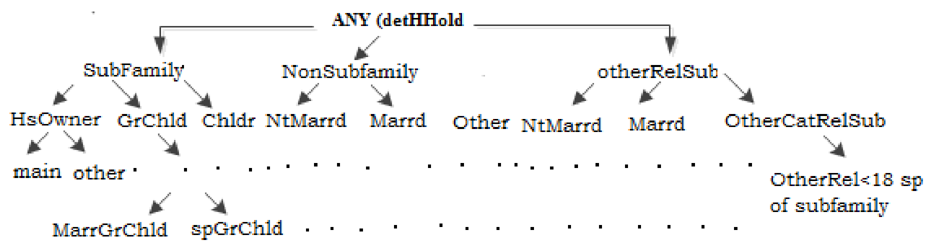
## Appendix E Table of Notation

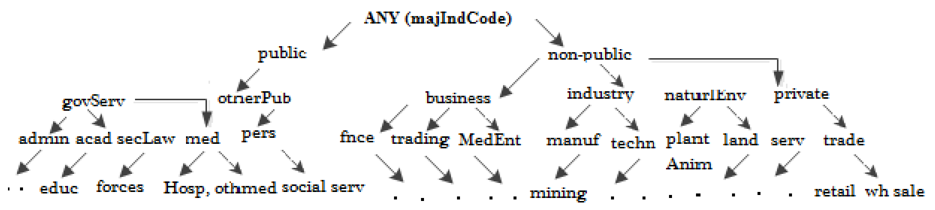
| Term | *Meaning* |
|---|---|
| Sig, $e_i, \nabla$ | *Significance or entropy of an attribute* |
| $\tau(C_i, A)$ | Tightness of a cluster C of attribute $A$ |
| Thr | Threshold (G.Thr, L.Thr) – Global and local threshold |
| $I_1^k$ | Intra cluster tightness in $k$ clusters (heuristic value 1 |
| $I_2^k$ | Inter cluster tightness in $k$ clusters (heuristic value 2) |
| $I_3^k$ | Cluster quality of $k$ clusters (heuristic value 3) |
| $v^k = l_i^k$ | Local attribute interestingness (harmonic aggregation) in iteration $k$ |
| $X^k$ | Local attribute heuristic values (1, 2 and 3) in iteration $k$ |
| $I_c$ | Interestingness linear function on concept value c |
| $H(I_c)$ | Interestingness non-linear function (probability density function – pdf) |
| $I_g^i$ | Global interestingness for attribute i (harmonic aggregation) |
| $I_g^D$ | Global interestingness for table D (harmonic aggregation) |
| $I_L^i$ | Total local cluster interestingness |
| $I_g^T$ | Total global cluster interestingness |

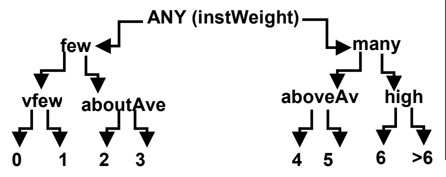
## Appendix F Concept Hierarchies

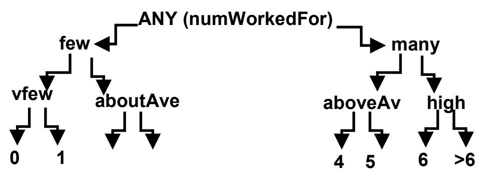


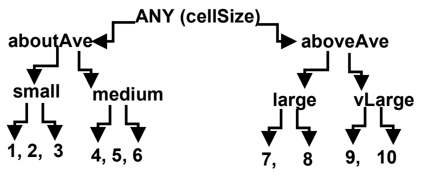


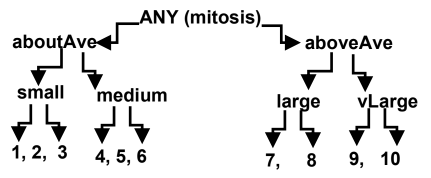

ANY (detHHold
SubFamily
NonSubfamily
otherRelSub
HsOwner
GrChld
Chldr
NtMarrd
Marrd
Other
NtMarrd
Marrd
OtherCatRelSub
main
other
OtherRel<18 sp of subfamily
MarrGrChld
spGrChld

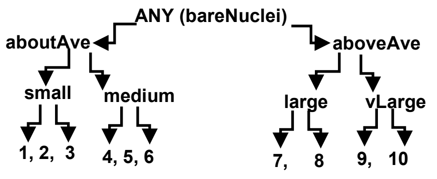

ANY (majIndCode)
public
non-public
govServ
otherPub
business
industry
naturlEnv
private
admin
acad
secLaw
med
pers
fnce
trading
MedEnt
manuf
techn
plant Anim
land
serv
trade
educ
forces
Hosp, othmed social serv
mining
retail
wh sale

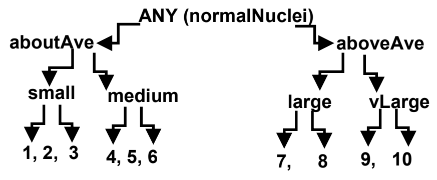

ANY (instWeight)
few
many
vfew
aboutAve
aboveAv
high
0
1
2
3
4
5
6
>6


ANY (numWorkedFor)
few
many
vfew
aboutAve
aboveAv
high
0
1
4
5
6
>6


ANY (cellSize)
aboutAve
aboveAve
small
medium
large
vLarge
1, 2, 3
4, 5, 6
7,
8
9,
10


ANY (mitosis)
aboutAve
aboveAve
small
medium
large
vLarge
1, 2, 3
4, 5, 6
7,
8
9,
10


ANY (bareNuclei)
aboutAve
aboveAve
small
medium
large
vLarge
1, 2, 3
4, 5, 6
7,
8
9,
10


ANY (normalNuclei)
aboutAve
aboveAve
small
medium
large
vLarge
1, 2, 3
4, 5, 6
7,
8
9,
10