# HiCHO: Attributes Based Classification of Ubiquitous Devices

Shachi Sharma[1], Shalini Kapoor[1], Bharat R. Srinivasan[2], and Mayank S. Narula[3]

[1] IBM Research Laboratory, New Delhi, India
[2] Georgia Institute of Technology, Atlanta, United States
[3] Indian Institute of Technology, Kharagpur, India

**Abstract.** An online and incremental clustering method to classify heterogeneous devices in dynamic ubiquitous computing environment is presented. The proposed classification technique, HiCHO, is based on attributes characterizing devices. These can be logical and physical attributes. Such classification allows to derive class level similarity or dissimilarity between devices and further use it to extract semantic information about relationship among devices. The HiCHO technique is protocol neutral and can be integrated with any device discovery protocol. Detailed simulation analysis and real-world data validates the efficacy of the HiCHO technique and its algorithms.

## 1 Introduction

Ubiquitous Computing as envisioned by Mark Weiser [14], is a computing paradigm where interconnected computational devices are part of day to day human life and are used invisibly to accomplish a task. Users work with a wide range of heterogeneous devices. These devices can be desktop devices, mobile communicators, digital assistants, wrist watches, game consoles, consumer electronics (e.g., TVs, radios, and refrigerators), cars, sensors, smart meters, video surveillance equipments etc. In many cases, users interact with multiple heterogeneous devices simultaneously. For example, a morning alarm clock instructs coffee machine to start preparing coffee automatically, road sensors inform arrival of vehicles to traffic signal so that it turns on. The alarm clock, coffee machine, road sensors and traffic signal are altogether different types of devices communicating with each other. Hence, design and development of smart applications and systems in ubiquitous computing environment require these heterogeneous devices to discover, identify and communicate with each other.

Today, discovery and identification of the devices is limited by the fact that devices following same protocol standard can find and interconnect with each other. There are instances when (i) devices are to be searched based on criterion like location, functionality, owners, services, etc i.e. the attributes characterizing devices and not by any protocol standard (ii) not only the singleton device but a group of devices sharing some characteristics are to be discovered. For example, consider a smart building with thousands of devices installed by several vendors.

A. Puiatti et al. (Eds.): MobiQuitous 2011, LNICST 104, pp. 113–125, 2012.

This may include air-conditioners, light sensors, access controls, web cams, video surveillance equipments, elevator sensors, telephones, utility meters etc. Each vendor provides its own monitoring capability. Moreover, generic monitoring tools use ip addresses based identification method to act on the remote devices. There is a need to monitor all devices in the building and apply some operations on a set of devices based on their attributes. Suppose if power consumption goes high on a particular day then all the devices (irrespective of the make) in the smart building with high energy consumption are to be switched off; similarly in case of outside temperature falling low, all the air-conditioners in the building are to be instructed to increase temperature. Another example can be of smart grid comprising of sub-stations, voltage regulators, capacitors, meters, reclosers, transmission lines etc. In case of some emergency situation such as fire in some part of the grid, a set of devices in the grid are to be discovered based on attributes such as location so that all the affected devices can be turned off instantly. Hence, one of the requirement for smart applications in ubiquitous computing environment is to *design and develop methods for classifying devices into groups based on criterion like logical attributes.* Based on such classification, it is easy to apply rules and policies, provide secure access control to a group of devices and perform finer grained device management.

Several protocols like Bluetooth, Jini, UpnP enable devices to locate each other [2] [12]. Bluetooth allows finding a set of devices in immediate vicinity based on device type [2]. However, these protocols do not have capability to group devices based on logical attributes as they do for physical attributes. The standard Zigbee defines notion of clusters [15]. In Zigbee, a cluster is a message or collection of messages pertaining to a given application domain. The idea behind clusters is to provide reusability by abstracting clusters across several application domains and placing them in a library organized according to pre-defined functional domains (e.g., lighting, closures, HVAC). *A generic technique to classify devices into groups based on their characteristics is not present today.* There are expected to be 50 billion devices worldwide by 2020, this mechanism, brings in a new paradigm where devices are identified and discovered based on the attributes they support.

In this paper, we propose a technique HiCHO to classify the ubiquitous devices using their logical attributes. This technique is independent of any protocol standard and can be implemented for all protocols. Another important point of discussion is that the set of logical attributes might change with time. For example, for a video camera at traffic signal, measurement of number of vehicles passing through is an important attribute whereas the model of video camera or the RAM it possesses to store images is not important for certain set of policies. Also over a period of time as devices within the vehicle itself become advanced, the measurement of number of cars, vis a vis truck would become increasingly important. Hence there is need to keep the classification mechanism open to add newer attributes.

We believe HiCHO can be used by several enterprises as they create their own intranets of things. With an increase in volume of ubiquitous devices in coming

years, a need will arise to control such devices beyond their physical attributes alone. The paper is organized into four sections. Section 2 discusses in detail the HiCHO method. This method is evaluated through simulated devices and also for real-world devices in section 3. The last section 4 contains concluding remarks.

## 2 HiCHO: Classification Technique for Ubiquitous Devices

Devices in real world have many characteristics such as color, manufacturer, type, owner, number of components, memory etc often known as their attributes. Following [3] [1] attribute and value can be defined as follows,

**Definition 1.** An Attribute *is a category in which a device can be classified such as 'color'.*

Attributes can be physical such as 'model', 'make' or logical such as 'owner', 'location'.

**Definition 2.** A Value *is the device's classification within that category, for example, 'red.'*

Attributes and values are free-form strings. Together, an attribute and its associated value form an attribute-value pair or av-pair. A set of av-pairs describes a device. We define,

**Definition 3.** Characteristics Set (CS) *as the set of all the possible av-pairs possessed by a device.*

For example, a mobile phone may have its $CS$ as {(manufacturer, 'Nokia'); (model, 'N72'); (color, 'black'); (wifi, 'no'); (camera, 'yes'); (owner,'john')}. In ubiquitous computing domain, we can assume that the universal attributes set $A$ is finite and is known in advance. Such an assumption is not too stringent to make as one can think of a scenario where the vendors and manufacturers of devices provide the information about attributes to standardizing agency whenever they launch a new device or upgrade an existing one. The other option to exempt this assumption is by learning the universal attributes set. It means whenever a device specifies a new attribute, the same is added to the universal attributes set automatically by the system implementing HiCHO technique. However, determining the value domain of an attribute may be difficult in many cases, for example, IP address. Hence, we do not make any assumption on the value set.

There are multiple ways to classify devices. One possibility is on the basis of physical attributes like device type, manufacturer ID, serial number, memory, hardware configuration. Such classification has limitations because - (i) physical attributes fail to provide a high level picture of importance of devices. The administrators instead have to find and feed this information. (ii) Many times, physical attributes are not known to the outside world, especially, human users. Some network tools and probes can retrieve this information but these tools themselves may give inaccurate and incomplete information. (iii) If there is some

```
<?xml version="1.0" ?>
  <!-- Sample XML Document -->
  <ns:deviceData xmlns:ns="urn:foo">
    <ns:attribute id="1">
      <ns:name>deviceType</ns:name>
      <ns:value>MobilePhone</ns:value>
    </ns:attribute>
    <ns:attribute id="2">
      <ns:name>manufacturer</ns:name>
      <ns:value>Nokia</ns:value>
    </ns:attribute>
   <ns:attribute id="3">
      <ns:name>model</ns:name>
      <ns:value>N72</ns:value>
    </ns:attribute>
   ...
   ...
   ...
   </ns:deviceData>
```

**Fig. 1.** XML document for specifying $CS$ : mobile phone example

change or enhancement in physical attributes, for example, memory increase, CPU change, then devices class may change necessitating re-classification.

Another option could be of grouping the devices manually based on logical attributes. Obviously, manual method is not scalable for huge number of devices. Also, such a classification becomes incomplete and outdated as new devices are added, new attributes are added or existing attributes undergo change. Hence, it is essential to automate the classification process based on logical attributes.

There is an immediate questions how devices specify their $CS$. The next subsection discusses this.

### 2.1 Specification of Characteristics Set

There are numerous ways by which a device can specify its $CS$. We opt to choose XML (Extensible Markup Language) because of its wide usage and applicability. XML is a mature standard and is already in practice for mobile and embedded devices. Menten etal [9] have used XML in designing network device management tools. XML-based messaging system for enabling communication between mobile devices is implemented by Kangasharju etal [7]. XML security specification has been extended and implemented for mobile devices [8]. The usage of XML has increased to the extent that now there are specialized devices in the network that control and manage XML traffic. The World Wide Web Consortium (W3C) has introduced a compression standard (Efficient XML Interchange i.e. EXI Format 1.0 [4]) for XML to bring the web data exchange standard to smart-phones and other power constrained mobile and embedded devices. Figure 1 shows the sample XML document of a mobile phone $CS$. Note that each attribute is a tuple - (name, value).

### 2.2 Clustering Devices

Many devices are almost identical to each other and share many similar av-pairs. There can be applications which may be interested in class level information or

which are developed for a particular class. Hence, there is a need to classify devices. We resort to clustering methods to partition devices into classes (called clusters) such that (i) devices that belongs to same cluster are similar in some ways (ii) devices that belongs to different clusters are dissimilar in some ways. Clustering is an extensively studied field and is widely used in domains like machine learning, data mining, bio-informatics, databases and statistics. *However, to the best of our knowledge, clustering is neither studied nor employed in the ubiquitous computing domain for classifying devices.* The ubiquitous computing devices have some specific requirements that a clustering method based on logical attributes must adhere to - (i) **on-line:** Since clustering method is to be based on logical attributes possessed by devices, values of these logical attributes may change over time. In some cases, even the attributes may undergo change. Also, clustering should allow devices to register and get classified in a cluster and de-register for de-allocating its class. This is important as in ubiquitous computing domain complete shutdown of devices is unacceptable if some hardware dependent softwares or application softwares are upgraded. (ii) **incremental:** The complete universe of the devices is not known in advance. Hence, clusters are built incrementally over time. This is in contrast to typical data mining problems where complete data set is available beforehand. (iii) **attributes and their values based:** The classification of devices in different groups must be such that while retrieving devices from a large population, it should not consume time. Devices can be searched by specifying complete or partial $CS$. The search of devices based on input criterion is time critical. If devices are first classified based on attributes then the search domain gets squeezed instantly. Then within those classes, av-pairs are used to find the final answers. Hence, the proposed method is *hierarchical in nature with two levels.* (iv) **capable of working with numeric as well as categorical data:** Attributes are free-form strings but values can be strings or numeric. The clustering method must be able to handle both types of values. The existing literature on clustering addresses the requirements mentioned above individually and not as a complete family. Work by Friedman [5] on attribute based clustering requires complete data set to be available. The clusters are created by calculating the distance between two data points and hence this method works only for numeric data. Real-time clustering algorithm (real-time OPTICS), by Shao etal [10] and stream data clustering by Shuyun etal [13] requires some training data to be available before deploying the algorithm in the environment. Real-time OPTICS is density based clustering method where clusters are constructed around core objects. This method is based on core-distances and hence applies only to numeric data set. Stream data clustering by Shuyun etal [13] first identifies micro-clusters using entropy measures, the online process of updating the micro-clusters is initiated afterwards where these micro-clusters represent the current snapshot of clusters which change over the course of the stream as new points arrive. There are specialized clustering algorithm for categorical data [11] but suffers from short comings like been non real-time and not incremental.

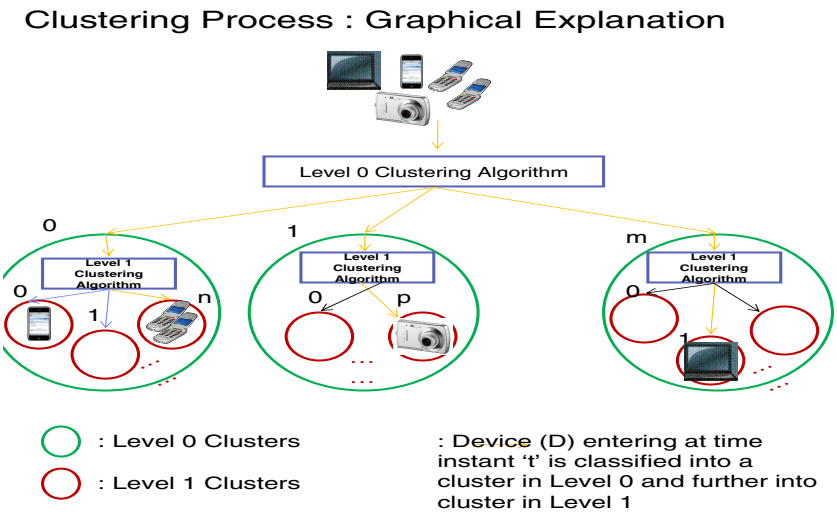


**Fig. 2.** Overview of HiCHO Technique

We propose a hierarchical (with two levels) clustering method - Level0 clustering based on attributes and Level1 clustering based attribute-values. As mentioned above, the reason behind designing a two step hierarchical method lies in reducing the search time. The overview of the HiCHO method is shown in Figure 2. We next present the methodology of these two steps in detail.

### 2.3 Level0 Clustering

The main idea of level0 clustering is to group together the devices whose attribute set is as close to each other as possible. The devices in different clusters have attribute sets as different from each other as possible. Consider two devices $D_i$ and $D_j$ with their attribute sets as $A_i$ and $A_j$ respectively where $A_i \subseteq A$ and $A_j \subseteq A$ where $A$ is the universal attribute set and is known in advance. One possible measure to compute similarity between two attribute sets is by using Jaccard Index [6]. Let $C_{ij}$ be the count of common attributes and $U_{ij}$ be the count of uncommon attributes between devices $D_i$ and $D_j$, then we define an index, termed as Commonality Index ($CI$) (Jaccard Index [6]) between $D_i$ and $D_j$ as,

$$CI(D_i||D_j) = \frac{C_{ij}}{C_{ij} + U_{ij}} \tag{1}$$

We also define cluster representative or $CR$ as,

**Definition 4.** A Cluster Representative or CR *is a virtual device whose attribute set is the union of attribute sets of all the devices belonging to that cluster.*

**Algorithm 1.** Level0 Clustering Algorithm

**Input**: $C, Re, \varepsilon_0, D_{new}, A_{new}, m$
**Output**: probably a new $C$ and $Re$

$max = 0$;
$cluster_{num} = 0$;
$tempCI[m]$; //a dummy array of size equal to number of clusters to store CI temporary
$i = 0$;
**if** $(C = \phi)$ **then**
    create a new cluster $C_{new}$;
    assign $D_{new}$ to $C_{new}$;
    $C = C \cup C_{new}$;
    create a new cluster representative $R_{new} = A_{new}$;
    $R = R \cup R_{new}$;
    return;
**while** $(i < m)$ **do**
    Calculate $C_{newi}$ and $U_{newi}$;
    $CI(D_{new}||Re_i) = C_{newi}/\,(C_{newi} + U_{newi})$;
    tempCI[i] = $CI(D_{new}||Re_i)$;
    i++;
//compute maximum of the array values and returns maximum value and cluster index
$cluster_{num} = MAXIMUM(tempCI, max)$;
**if** $(max \geq \varepsilon_0)$ **then**
    assign $D_{new}$ to cluster $C_{cluster_{num}}$;
    $Re_{cluster_{num}} = Re_{cluster_{num}} \cup A_{new}$;
**else**
    create a new cluster $C_{new}$;
    assign $D_{new}$ to $C_{new}$;
    $C = C \cup C_{new}$;
    create a new cluster representative $R_{new} = A_{new}$;
    $R = R \cup R_{new}$;

The basis of level0 clustering algorithm are *CI* and *CR*. Let there are $m$ clusters and $C = \{C_1, C_2, ..., C_m\}$ be collection of all the clusters and $Re = \{Re_1, Re_2, ..., Re_m\}$ be corresponding collection of cluster representatives at any time $t$ at level 0 when a new device $D_{new}$ enters. Let $A_{new}$ be the attribute set of device $D_{new}$. Then *CI* is calculated between $D_{new}$ and cluster representatives of all the $m$ clusters. $D_{new}$ is allocated to the cluster with maximum commonality i.e. the one with maximum value of *CI*. The cluster representative of selected cluster is then expanded to include new attributes. The complete algorithm is presented in Algorithm 1. A new cluster is created if $CI(D_{new}||Re_k) < \varepsilon_0,\ \forall\ k = 1, 2, ..., m$ i.e. $D_{new}$ does not have any significant commonality with any of the existing cluster. When the very first device enters, there is no cluster and a new cluster with first device is created. Here, $\varepsilon_0$ is the level0 clustering parameter and controls the size and number of clusters. We discuss the sensitivity of level0 clustering algorithm with $\varepsilon_0$ in section 3.

### 2.4 Level1 Clustering

Level1 Clustering is applied on each of the clusters of level0. This clustering is based on av-pairs. Here, we assume that attributes have some weights attached to them. The weights of an attribute may be its importance, its dispersion in the devices population or entropy. Let $W = \{w_1, w_2, ..., w_N\}$ be set of weights

**Algorithm 2.** Level1 Clustering Algorithm

**Input**: $C_{x1}, \varepsilon_1, D_{new}, A_{new}, nx$
**Output**: probably a new $C_{x1}$

```
max = 0;
cluster_num = 0;
tempAvgSI[nx]; //a dummy array of size equal to number of clusters to store SI temporary
tempSI[]; // another dummy array to store intermediate values of SI
i = 0;
j = 0;
if (C_x1 = φ) then
    create a new cluster C_new;
    assign D_new to C_new;
    C_x1 = C_x1 ∪ C_new;
    return;
while (i < nx) do
    while (j < |C_x1_i|) do
        tempSI[j] = SI(D_new||(D_j ∈ C_x1_i));
    tempAvgSI[i] = SUM (values of tempSI)/|C_x1_i|;
//compute maximum of the array values and returns maximum value and cluster index
cluster_num = MAXIMUM(tempAvgSI, max);
if (max ≥ ε_1) then
    assign D_new to cluster C_cluster_num;
else
    create a new cluster C_new;
    assign D_new to C_new;
    C_x1 = C_x1 ∪ C_new;
```

of attributes where $w_i$ is weight of attribute $a_i$. These weights are normalized weights,

$$\sum_{i=1}^{N} w_i = 1 \tag{2}$$

Between two devices $D_i$ and $D_j$, we compute following quantities:

1. Find the set of same av-pairs i.e. between characteristic set of $CS_i$ of device $D_i$ and $CS_j$ of device $D_j$ calculate

$$Z_{ij} = CS_i \bigcap CS_j \tag{3}$$

2. Between $D_i$ and $D_j$, find the set of those attributes $a_k$ such that
$a_k \in \left\{ Z'_{ij} = A_i \cap A_j \right\}$ and
$\forall a_k \in Z'_{ij}, \quad v_{a_k}(i) \neq v_{a_k}(j)$
where $v_{a_k}(i)$ is the value of attribute $a_k$ for device $D_i$.

We then define a Similarity Index (SI) between $D_i$ and $D_j$ as,

$$SI(D_i||D_j) = (\alpha + \beta) \sum_{av_k \in Z_{ij}} w_{a_k} + \alpha \sum_{a_k \in Z'_{ij}} w_{a_k} \tag{4}$$

Here, $\alpha > 0$ and $\beta > 0$ are two parameters that controls the importance given to attributes and av-pairs respectively. One obvious thing is to chose $\alpha < \beta$, assign more importance when av-pair is same. In order to keep $SI$ normalized, a required condition is $\alpha + \beta = 1$.

Let a new device $D_{new}$ finds $C_{x1} = \{C_{x1_1}, C_{x1_2}, ..., C_{x1_{nx}}\}$ $nx$ level 1 clusters within level 0 cluster $C_x \in C$ upon its arrival. For a level0 cluster $C_{x1_i}$, $SI$ is calculated between $D_{new}$ and all the members of $C_{x1_i}$. The mean value of $SI_{mean}$ for that cluster is then computed. The process is repeated for all the $nx$ clusters. The device $D_{new}$ is then allocated to that cluster from $C_{x1}$ which has maximum value of $SI_{mean}$ and is greater than the clustering parameter $\varepsilon_1$. The level1 clustering algorithm is presented in Algorithm 2.

## 3 Performance Evaluation of HiCHO

In this section, we discuss the performance results of our proposed HiCHO technique. Our principal objective here is (i) to demonstrate the scalability of the HiCHO algorithms through simulation (ii) to study the sensitivity of clustering algorithms with respect to clustering parameters $\varepsilon_0$ of level0 and $\varepsilon_1$ of level1 (iii) to show the applicability of the HiCHO for real-world devices. We focus on two performance metrics : (i) average number of clusters created at each level and (ii) the average size of clusters i.e. the average number of devices in a cluster at each level.

We have implemented a simulator that generates characteristics set of devices in XML (as depicted in Figure 1). The attributes from the universal attributes set are chosen following Uniform and Pareto distribution. Uniform distribution is an obvious choice as it corresponds to an extreme case where all attributes have equal probability to be possessed by the devices. In practice, some attributes tend to occur in a large number of devices (e.g. color) than others (e.g. service). This behavior can be modeled by choosing a skewed distribution for selecting attributes. We opt for Pareto distribution because of its wider usage in modeling skewed behavior. The performance of level0 algorithm of HiCHO is shown Figures 3. As depicted in Figure 3 (a), Uniformly distributed *CS* devices result in more number of clusters compared to Pareto distributed *CS* devices. This is because of the way *CS*'s are generated by the simulator. For Pareto distributed *CS* devices, the fewer number of attributes are selected repeatedly whereas for Uniformly distributed *CS* devices all the attributes are chosen. For a fixed value of $\varepsilon_0$ and maximum size of *CS*, the number of clusters increase sharply initially as devices enter and then almost becomes constant. The average size of the cluster representatives also remains constant with number of devices as illustrated in Figure 3 (b). The average size of clusters increases gradually with increase in number of the devices as presented in Figure 3 (c). These facts shows scalability of the level0 clustering algorithm. Figures 3 (d)-(f) present sensitivity of level0 clustering algorithm with clustering parameter $\varepsilon_0$ for fixed number of devices and maximum *CS* size of 10. Lower values of $\varepsilon_0$ results in small number of large clusters whereas high value of $\varepsilon_0$ gives rise to large number of small clusters.

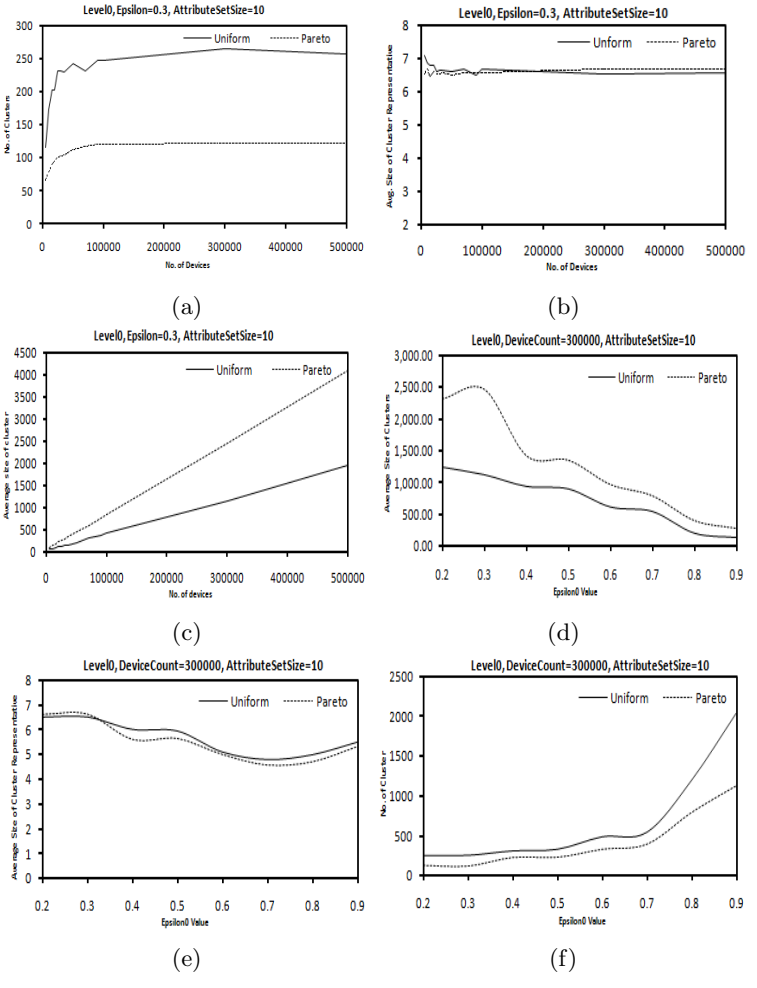


**Fig. 3.** Performance of Level0 Clustering Algorithm of HiCHO

Hence, fixing $\varepsilon_0$ to middle range is expected to give a balance between number of clusters and cluster sizes. Level1 Clustering is carried out within each cluster created at Level0. The performance results of level1 clustering algorithm is shown in Figures 4. Figure 4 (a) and (b) show that high value of the parameter $\beta$ compared to $\alpha$ results in large number of small clusters for Uniformly distributed *CS* devices. Whereas for Pareto distributed *CS* devices, the choice of parameters $\alpha$ and $\beta$ have very little impact on number of clusters and cluster sizes as is clear from Figure 4 (c) and (d). For Uniformly distributed *CS* devices, both the number of clusters and cluster size becomes constant as the clustering parameter $\varepsilon_1$ tends to one. This phenomenon is not observed for Pareto distributed *CS* devices. In order to validate the feasibility of the HiCHO to real-world devices, we have gathered the attribute-value data of 20 devices owned by

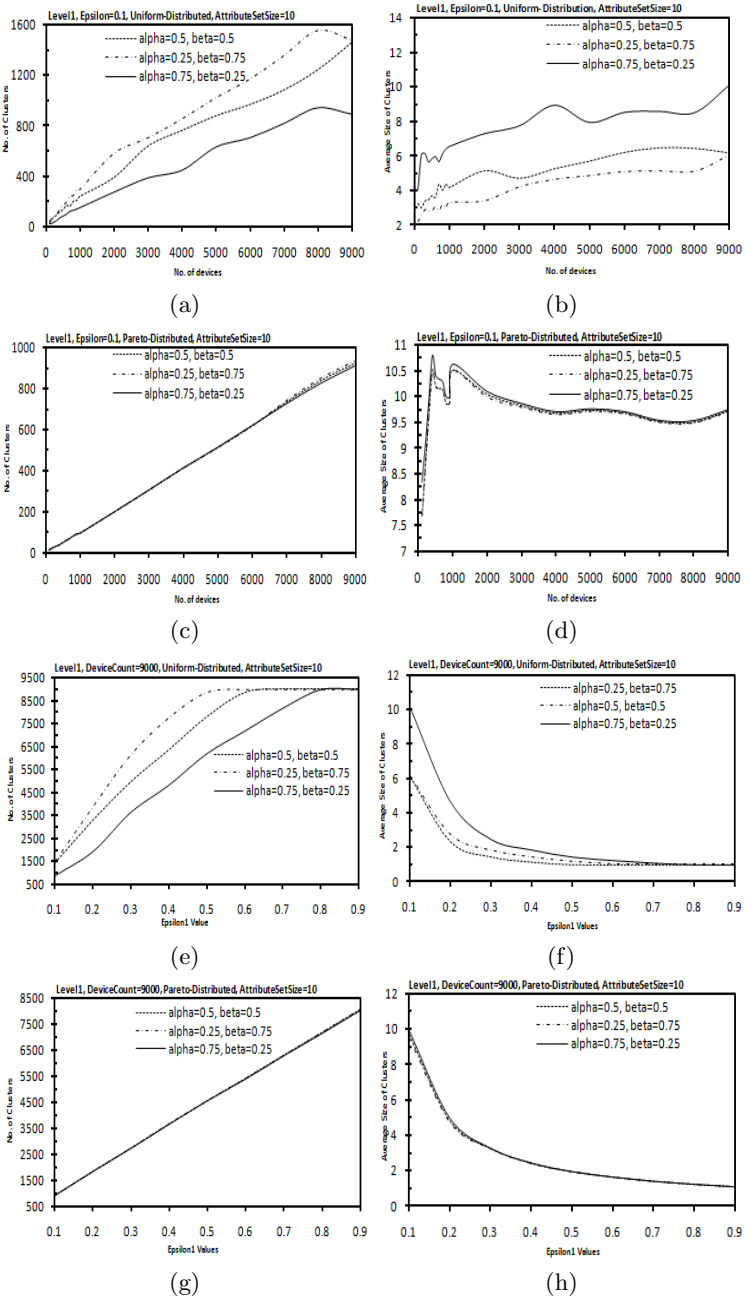


**Fig. 4.** Performance Results of Level1 Clustering Algorithm of HiCHO

our colleagues at the IBM. These devices are laptops, mobile phones and digital cameras. These 20 devices have maximum of 18 attributes including pixels,

**Table 1.** Level0 Results of Real world Devices

| Cluster id | Cluster Size |
|---|---|
| 0 | 4 |
| 1 | 7 |
| 2 | 8 |
| 3 | 1 |

**Table 2.** Level1 Results of Real world Devices

| Cluster id | Cluster Size |
|---|---|
| 01 | 3 |
| 02 | 1 |
| 11 | 2 |
| 12 | 3 |
| 13 | 2 |
| 21 | 2 |
| 22 | 3 |
| 23 | 3 |
| 31 | 1 |

owners, model, functional capabilities etc. We fix the level0 clustering parameter $\varepsilon_0$ to 0.6 and level1 clustering parameters $\alpha$ and $\beta$ to 0.5; $\varepsilon_1$ to 0.45. We then run the HiCHO methods on characteristics set of these 20 devices and results are presented in Table 1 and 2. In spite of the fact that the HiCHO method is applied to a small-scale real world data set, it allows us to understand the clustering pattern in practice. Level0 of HiCHO classifies the 20 devices into 4 classes. The formation of clusters is not based on single attribute such as device type or color rather the combination and presence of attributes govern the creation of clusters at level0. The second step of level1 clustering fine grains clusters within level0 clusters based on both attributes and values. We observe that all the devices of same color and make are combined, or devices belonging to same owner are clubbed together into one cluster, or devices with same protocol like bluetooth are grouped together at level1.

## 4 Conclusion and Future Work

A classification technique, HiCHO, based on attribute-value pairs of devices is proposed. These attributes are logical including capabilities, functionalities, customized data etc. Devices specify the attribute-value pairs, called Characteristics Set ($CS$), in XML. In practice, we either expect $CS$ to be available with devices through some configuration or network probes to obtain and transfer $CS$ to back-end server running HiCHO algorithms. The method itself is not tied to any protocol and hence can be integrated with any device discovery protocol. This type of classification is useful in generating class-level logical identifiers and in creating repository of devices for future networks like Internet of Things (IoT). Though we have explained HiCHO in relation to logical attributes, but it can be very well applied to a combination of physical and logical attributes. This will become important for IoT where objects needs to be classified by a combination of physical and logical attributes. So far, we have tested HiCHO for a small set of real-world devices to show the feasibility of the technique. As a future work, we would like to establish the scalability of HiCHO method to a large set of real-world devices.

## References


1. Adjie, W.: The design and implementation of an intentional naming system. In: 17th ACM Symposium on Operating Systems Principles, pp. 186–201 (1999)
2. Blueetooth website, http://www.bluetooth.com/
3. CCITT. The Directory Overview of Concepts, Models and Services, X.500 series recommendations (1988)
4. Efficient XML Interchange, http://www.w3.org/TR/exi
5. Friedman, J.H.: Clustering objects on subsets of attributes. Journal of Royal Statistical Society 66(4), 815–849 (2004)
6. Jaccard, P.: Étude comparative de la distribution florale dans une portion des Alpes et des Jura. Bulletin de la Société Vaudoise des Sciences Naturelles 37, 547–579 (1901)
7. Kangasharju, J., Lindholm, T., Tarkoma, S.: Xml messaging for mobile devices: From requirements to implementation. Computer Networks: The International Journal of Computer and Telecommunications Networking 51(16) (2007)
8. Kangasharju, J.: Efficient Implementation of XML Security for Mobile Devices. In: IEEE International Conference on Web Services, pp. 134–141 (2007)
9. Menten, L.E., Murray, H.: Experiences in the application of xml for device management. IEEE Communications Magazine 72(7), 92–100 (2004)
10. Shao, F., et al.: A new real-time clustering algorithm. Journal of Information and Computational Science 7(10), 2110–2121 (2010)
11. Li, T., Ma, S., Ogihara, M.: Entropy-based criterion in categorical clustering. In: Proceedings of the 12th International Conference on Machine Learning (2004)
12. UpnP, http://www.upnp.org/
13. Wang, S., et al.: Entropy based clustering of data streams with mixed numeric and categorical values. In: IEEE/ACIS International Conference on Computer and Information Science, pp. 140–145 (2008)
14. Weiser, M.: The Computer for the Twenty-First Century. Scientific American, 94–104 (1991)
15. Zigbee website, http://www.zigbee.org/