A Model for Quality Guaranteed Resource-Aware Stream Mining

Marcel Karnstedt[1] Conny Franke[2] Mohamed Medhat Gaber[3]

[1] Technische Universität Ilmenau, Ilmenau, Germany
[2] University of California at Davis, Davis, CA, USA
[3] Tasmanian ICT Centre, CSIRO ICT Centre, Australia

Abstract. Data streams are produced continuously at a high speed. Most data stream mining techniques address this challenge by using adaptation and approximation techniques. Adapting to available resources has been addressed recently. Although these techniques ensure the continuity of the data mining process under resource limitation, the quality of the output is still an open issue. In this paper, we propose a generic model that guarantees the quality of the output while maintaining efficient resource consumption. The model works on estimating the quality of the output given the available resources. Only a subset of these resources will be used that guarantees the minimum quality loss. The model is generalized for any data stream mining technique.

1 Introduction

In the past years, data streams emerged as a new kind of data source. Analyzing data streams thus becomes more and more important as new areas of application are identified. Applications like click stream analysis and the analysis of records from networking and telephone services are among the most popular examples for data stream mining, i.e., the discovery of patterns and rules in the data. Another important area of application is the stream processing in sensor networks, where continuously generated data is processed as far as possible onboard the sensor node in order to preserve the limited bandwidth and energy.

Due to the unique characteristics of data streams, like their potentially infinite nature and the vast amount of data they are carrying, data stream mining requires a different processing than mining on databases and data warehouses. Efficient resource consumption is one of the major objectives when designing stream mining algorithms. Rather than storing the incoming data and processing it offline like in traditional data mining, data stream mining is much more constraint in terms of available resources.

Most data stream algorithms provide approximate results, often by using a summarization of the stream (called a *synopsis*) and determining precise error bounds. Thus, a notion of output quality is immediately associated with this process. Which information from the data stream is stored is crucial for the quality of the data mining results. Note that we explicitly refer to the output

quality of the mining technique, in contrast to aspects of the input quality, which is a related but still different field of research.

Well-known state-of-the-art of stream mining algorithms reveal that while many of the algorithms strive for minimizing the resources they need, most of them are not designed with regard to adaptation to resource availability. Specifically, they fail to provide well-defined routines for situations where the available resources are exhausted. Most algorithms are designed to work in a static manner, without taking into account that the algorithm's resource requirements might exceed the amount of resources provided. In such a case the algorithm's behavior depends in its implementation, and thus might be undefined. Recent approaches (e.g., [1]) identified this issue, but still lack a clear consideration of correlations between resource-adaptation and output quality.

When dealing with complex stream mining systems, where usually a set of queries runs continuously and resources are shared among them, we additionally have to consider the interactions between different mining operators. In such systems, algorithmic output quality is usually referred to as Quality-of-Service (QoS) [2]. Note that, when combining resource and quality aspects of multiple, possibly dependent, mining operators, we start closing the mentioned gap between input quality and output quality of mining techniques.

In this work, we consider all of the aforementioned aspects and integrate them into one single framework. We propose a generic three layer model for quality guaranteed resource-aware (QGRA) data mining on data streams. The model is designed to be applicable to a wide variety of stream mining techniques. Our model assesses the output quality and the current status of resources and adapts the algorithm's resource consumption accordingly. This way, we are able to maintain resource efficiency and we may use lesser resources to achieve the same level of accuracy in the output. At any time, we will be having the maximum achievable quality according to the resources. Most static data stream mining algorithms leave excess resources unused. With our framework, available resources are utilized in an optimal way at any point in time.

The model utilizes a set of functions that is provided by the applied algorithm to control the adaptation. One function is used to determine the algorithmic parameters from the assessed resources. Then, a second function is used to determine the output quality based on the chosen algorithmic parameters. Other functions are used to compute lower bounds for the algorithmic parameters in order to maintain the quality of the output. Using this strategy, our model bridges the gap between quality-aware mining and general resource-adaptivity in data stream mining by monitoring the resource consumption.

The remainder of this paper is organized as follows. In Section 2 we provide some background about data mining quality and resource-aware data stream processing. Our formal model is introduced in Section 3. There, we give a brief description of the functionalities and a formalization of all elements. Finally, Section 4 concludes this paper and outlines areas for future work.

2 Background

2.1 Data Mining Quality

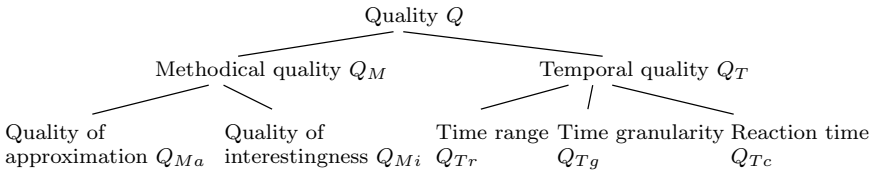

Fig. 1. Different quality measures

Most data mining algorithms have control parameters to determine how well their output approximates the actual result. These control parameters differ for each technique and data mining goal, but they can be arranged into classes of related parameters. Typical examples include the number of microclusters maintained during clustering, or the maximum frequency error in frequent itemset mining. The values of these parameters have a strong influence on the workload of the algorithm as well as on the size of the synopsis it maintains. In general, the better the approximation of the output should be, the more resources the algorithm consumes. Due to this close correlation between these parameters and the output quality of an algorithm, we will refer to this set of parameters as *adaptation factors*. After identifying the adaptation factors of an algorithm, we are able to adapt its resource requirements and output quality.

In analogy to a classification of adaption factors, affected quality measures can be classified, too. We distinguish several different classes of quality measures, which are categorized in Figure 1. This classification is comprehensive, yet extensible without restricting the proposed framework. All Q_{T*} are identical for different mining problems and symbolize concrete quality measures, while Q_{M*} represent classes of measures that are always specific to the investigated problem and the applied algorithm(s). For example, in the context of clustering these measures involve the clustering quality, e.g., SSQ, diameter and other standards to evaluate the final result of a clustering. One traditional measure for the problem of frequent itemset mining is the error rate ϵ, which defines the maximal deviation of the observed frequency to the actual frequency of an itemset. For several specific mining applications, special interestingness measures (Q_{Mi}) have been proposed in the literature (e.g., [3]). In the context of frequent itemsets the support is one such interestingness measure.

As many existing algorithms take time sensitiveness into account, we define time as another important quality measure. Q_{Tr} describes how far we can look back into the history of the processed data stream and Q_{Tg} how exact we can do this, which means which time granularity we can provide. Q_{Tc} corresponds to one of the main challenges of stream mining: the actual time necessary to register changes in the stream. As might be expected, adaptation factors sometimes

influence more than one of these quality measures, making them dependent from each other. For more details on the different quality classes we refer to [4]. For the remainder of this paper, if we refer to all quality measures as a whole, we will use the symbol of the superclass Q and the general term 'quality'.

2.2 Resource and Quality Awareness

Most data stream mining algorithms are designed to use as little resources as possible. However, they are often not aware of the actual amount of resources available and thus may either fail to utilize them completely or may not be able to work properly with the given resource constraints. We therefore distinguish algorithms that are aware of the available resources and are able to adapt their requirements accordingly. When talking about resources, we do not only consider memory consumption, despite this being the main factor in most streaming applications. In addition, since data streams are often generated at a rapid rate, algorithms must need only minimal time to process the data in order to keep up with the pace of the stream. In the example of sensor networks, bandwidth and battery power are additional constraint resources.

Apart from the adaptation factors, properties of the data stream also have a strong impact on an algorithm's resource requirements. One of the most important properties is the streaming rate. Another one are characteristics of the individual elements in the data stream, like the range and distribution of their values, and their size. This correlates to adaptation methods like sampling and load shedding [5], which can be used on the input level of mining techniques to reduce the workload, and thus the resource requirements, by decreasing the volume of the incoming stream. Due to their generic nature, they are applicable to all mining algorithms, since they do not require any changes to the algorithm itself. As a consequence, however, applying these methods results in the loss of guaranteed error bounds which the original algorithm may have provided. Instead, only probabilistic error bounds can be established. This may be a non-desirable tradeoff for some applications. Moreover, determining these probabilistic bounds should be expected to be very complicated and, due to the evolving character of streams, potentially erroneous.

Other levels of resource-adaptation throughout the whole process of stream mining have been identified and discussed in recent works. Methods that can be applied to most data stream mining algorithms have been proposed for example in [6]. Most other approaches either do not formalize their approaches accordingly or focus on single, rather limited, levels of adaption. Moreover, to the best of our knowledge, all of them lack the combination of resource and quality awareness. That means, although they deal with resource adaptation, they do not take quality aspects and guarantees into consideration.

3 QGRA Model

3.1 Model Description

Gaber et al. [6] have proposed and developed a generic model to adapt data stream mining algorithms to the current availability of computational resources. The model aims at prolonging the life-time of the running technique in critical situations of low availability of computational resources. It has been coined as *Algorithm Granularity (AG)*. AG can adapt the consumption of computational resources according to measured patterns of consumption over a pre-set time window.

AG has been classified into three classes according to the adaptation end-points. *Algorithm Input Granularity (AIG)* adapts the input streaming data to the mining algorithm. On the other hand, Algorithm Output Granularity (AOG) changes the output size of the algorithm. Finally, *Algorithm Processing Granularity (APG)* can adapt the algorithm parameters to consume less CPU cycles. The changes in AG affect the accuracy of the output. Therefore, the model sets bounds on the AG settings in order to keep the accuracy loss bounded. These AG settings have been used to develop an online clustering algorithm termed as *Resource-Aware Cluster (RA-Cluster)*.

Although the AG model has proven its applicability to change the resource consumption, the model is still facing the following issues:

- The bounds over the AG settings have no guarantee over the quality of the output. Because the quality relies on many other interleaving factors such as data distribution and the running mining technique.
- The changes in the AG settings are not quality-aware. That means the algorithm changes according only to the availability of computational resources. This may lead to accuracy loss and/or extra use of computational resources, because in some cases, we can gain the same accuracy using less resources.
- The AG settings do not take into consideration the interaction among the different settings. Addressing this issue can optimize the use of resources.

In this paper, we propose a new model QGRA that extends AG in order to address the above issues. The model is able to adapt in real-time according to resource consumption patterns as well as the quality of output. Franke et al. [4] have proposed a quality-aware data stream mining model. This model will be extended to assess the quality of the output in real-time. This assessment will be used to choose the best combination of AG settings that minimize resource consumption, and maximize the quality of output.

The model has three layers. The first one is the resource monitoring that works over dynamic time intervals. Unlike the AG model, the time window is dynamic and changes according to the criticality of the available computational resources. The second component is the real-time quality assessment. This will be able to provide information about the quality of the output given the availability of resources. It will also be able to provide the system with information about preserving computational resources while maintaining the same quality

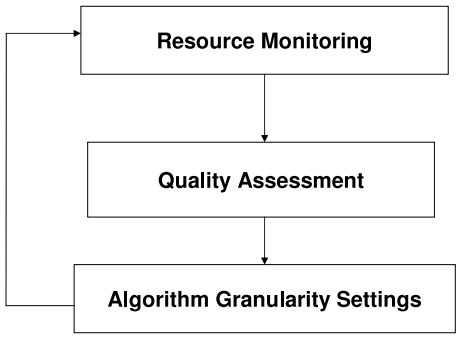

Fig. 2. Three layer model

level. The output of the second component will be passed to the AG settings (AGS) component. This third component feeds the mining algorithms with input, output and processing settings.

3.2 Formalization

The QGRA model relies on the notion of *variables* $v \in V$, which comprises all dynamic components the internal formulas and mechanisms are built on. We define a partitioning $\breve{V}$ over V, where each v belongs to a certain class $\hat{C}$ of variables, described as follows. As $\breve{V}$ is a partitioning,

$$\forall \hat{C}_1, \hat{C}_2 \in \breve{V}, \hat{C}_1 \neq \hat{C}_2 : \hat{C}_1 \cap \hat{C}_2 = \emptyset$$

holds. That is, each variable v belongs to exactly one class $\hat{C}$ of variables.

On the one side of the adaptation framework, we assume a set of *resources* $\hat{R} \in \breve{V}$,

$$\hat{R} := \{r | r \text{ is a limited resource consumed by the algorithm}\}.$$

Main representatives of $\hat{R}$ are the consumed memory and the number of CPU cycles needed. This corresponds to the kinds of resources already considered in previous works like [6, 4, 1].

The basis of the framework is enriched by quality awareness, defined in terms of *quality measures* $\hat{Q} \in \breve{V}$,

$$\hat{Q} := \{q | q \text{ is a quality measure of interest}\}.$$

There is a wide variety of quality measures that might be integrated into $\hat{Q}$. We present a brief classification in Section 2.1, which is based on the more detailed work in [4].

On the contrary side of the framework, we expect any stream mining algorithm to define a set of *parameters* $\hat{P} \in \breve{V}$,

$$\hat{P} := \{p | p \text{ influences resource requirements and/or output quality of the algorithm}\}.$$

These parameters can be seen as the "tuning knobs" of the particular method. This includes representatives of adaptation factors corresponding to the afore mentioned classes AIG, APG and AOG, e.g., sampling rate, internally used thresholds or number of output objects (like the number of output clusters). In addition, we also include parameters that cannot be adjusted but the method exhibits some dependence on, in either resource requirements, output quality, or both. Examples of this kind of parameters are distribution models the stream data follows or the fraction of noise. More on the specific requirements an algorithm has to meet can be found in Section 3.3.

In [6] the sets $\hat{R}$ and $\hat{P}$ were used in order to implement a one-way resource adaptation. This means, predictions for relevant $r \in \hat{R}$ are used to adapt $p \in \hat{P}$ accordingly, while the prediction of future resources is based on the observed recent resource consumption. Now, we introduce how to combine this idea with the quality-awareness proposed in [4].

We define an instance $F(\hat{C})$ as a materialization of all variables from class $\hat{C} \in \breve{V}$ which are actually involved in the adaptation process, i.e.,:

$$\begin{aligned} F &: \breve{V} \rightarrow V \times \Re \\ F(\hat{C}) &:= \{v, f(v) | v \in \hat{C} \wedge \nexists w \neq v : w \in \hat{C} \\ &\quad \wedge f(v) \text{ is the current value of } v\}. \end{aligned}$$

As v only defines which variable is concerned, $f(v)$ represents an actual value of that variable. To improve readability, we use C short for $F(\hat{C})$. In other words, C represents all variables from class $\hat{C}$ together with the corresponding values. For instance, we use R to represent all limited resources and their actual amount currently consumed by a specific algorithm.

Informally speaking, $\hat{C}$ represents the schema level of the variable classification, i.e., a description of which variables belong together in a class. Accordingly, C denotes an instance of $\hat{C}$, that is, it associates each variable $v \in \hat{C}$ with an actual value $f(v)$. In the above definition of C, we write $\nexists w \neq v : w \in \hat{C}$ to denote $\forall (v, f(v)) \in F(\hat{C}) : v \in \hat{C} \wedge \forall v \in \hat{C} : (v, f(v)) \in F(\hat{C})$. That is, on the instance level we have exactly one value $f(v)$ for each $v \in \hat{C}$. Note that distinguishing between classes $\hat{C}$ and instances C is not urgently necessary to make the model work. However, we believe that this makes the model more flexible, resulting in more algorithms and approaches being applicable to it. For instance, the introduced notion allows for an easy but still mathematically consistent integration of "schema-based" functions, e.g., which select actually considered quality measures from the set of all possible ones.

On the notion of the variables and instances we introduce two more kinds of sets. First, we define R_L to represent current resource limits and Q_L to represent requested quality guarantees. Whether these limits are met or not is described by two functions:

$$\Phi(R_L, R) := \begin{cases} \text{true} & \text{if } \forall (v, x) \in R_L : (v, y) \in R \wedge x \geq y \\ \text{false} & \text{else} \end{cases}$$

The resource limits R_L are met in R, i.e., $\Phi(R_L, R) = true$, if for each $v \in \hat{R}$ its value y in the instance R is less than or equal to its limit x in R_L.

$$\Psi(Q_L, Q) := \begin{cases} \text{true} & \text{if } \forall (v,x) \in Q_L : (v,y) \in Q \wedge x \leq y \\ \text{false} & \text{else} \end{cases}$$

The quality guarantees Q_L are met in Q, i.e., $\Psi(Q_L, Q) = true$, if for each $v \in \hat{Q}$ its value y in the instance Q is greater than or equal to its limit x in Q_L.

Finally, we define *timelined* variable instances C_T. A set C_T corresponds to an instance C enriched by a timestamp t, which represents the time the according values were effective. Thus,

$$\begin{aligned} C_T := \{v, x, t | (v,x) \in C \wedge\ & \nexists (w,y) \neq (v,x) : (w,y) \in C \\ & \wedge (v,x) \text{ was effective at time } t\}. \end{aligned}$$

C_T associates each pair $(v,x) \in C$ with a timestamp t. In the following, if we refer to an instance of one specific time t we write C_t for short.

On the introduced sets we define the following functions:

$$\rho : R_L \times R_T \times P_T \rightarrow \{-1, 0, 1\} \tag{1}$$
$$\phi : P \rightarrow R \tag{2}$$
$$\psi : P \rightarrow Q \tag{3}$$
$$\tau : R_L \times R_T \times P_T \rightarrow P \tag{4}$$
$$\omega : Q_L \times Q_T \times P_T \times P \rightarrow P \tag{5}$$

The first formula ρ is used to decide whether future resource consumption should be increased (underload situation, $\rho = 1$), decreased (overload situation, $\rho = -1$) or nothing is to be done at all ($\rho = 0$). This decision is based on provided resource limits, recent resource consumption and recent parameters. There are different approaches for handling this issue. [6] proposes to calculate the number of time frames remaining until resources are exhausted, whereas [4] uses a filling factor describing the percentage of available resources already consumed.

ϕ and ψ take as input an instance of parameters and map them to the resulting instance of resources and quality measures, respectively. τ and ω can each be seen as a kind of inverse function, mapping an instance of resources, respectively quality measures, to an instance of parameters. As additional input, both τ and ω accept recent parameter values and recent resources/quality measures. In order to align the quality-based decision with a preceding resource-based decision, ω also takes suggested parameters P as an input.

Based on these sets and functions the QGRA model works as illustrated in Algorithm 1.

As mentioned before, the time intervals in which the QGRA method is applied are set dynamically. In the beginning, all parameters are set to achieve highest possible quality. The algorithmic steps from Algorithm 1 are then applied at any time t. Based on the current resource consumption in time t provided

Algorithm 1 General QGRA algorithm at time t

1: $R_t = \text{res-mon}(t)$
2: **if** $0 \neq \rho(R_L, \cup_{i=1}^{t} R_i, P_t)$ **then**
3: $\quad P = \tau(R_L, \cup_{i=1}^{t} R_i, P_t)$
4: $\quad P' = \omega(Q_L, \cup_{i=1}^{t} Q_i, P_t, P)$
5: $\quad$ **if** $\Phi(R_L, \psi(P'))$ **then**
6: $\quad\quad P = P'$
7: $\quad$ **end if**
8: $\quad$ **if** $!(\Phi(R_L, \phi(P)) \wedge \Psi(Q_L, \psi(P)))$ **then**
9: $\quad\quad$ **return** false
10: $\quad$ **end if**
11: **else**
12: $\quad P = P_t$
13: **end if**
14: $Q_{t+1} = \{q, x, t+1 | (q, x) \in \psi(P) \wedge \nexists (u, y) \neq (q, x) : (u, y) \in \psi(P)\}$
15: $P_{t+1} = \{p, x, t+1 | (p, x) \in P \wedge \nexists (o, y) \neq (p, x) : (o, y) \in P\}$
16: set parameters P
17: **return** true

by the resource monitoring component (line 1) and (dynamically) provided resource limits, ρ is used to decide whether resource utilization should be increased, decreased or maintained (line 2). If any adaptation is necessary, a new set of parameters is determined using τ on the provided resource limits and on recent resource consumption as well as parameters (line 3). In the next step, we refine these parameters using ω on the QoS requirements and on recent quality measures as well as recent parameters (line 4). In order not to work contrary, ω also takes the set of parameters suggested before by the resource adaptation as additional input. Only if the parameters modified like this still meet the resource limits (line 5), they are accepted (line 6). After all adaptation steps, the new parameters are checked for resource and QoS requirements again (line 8). If they are not acceptable, the failed resource and/or quality requirements are signalized and reaction is left to the system or user. Otherwise, the resulting qualities and parameters are stored for later access (lines 14 & 15) and finally set (line 16). Note that, if no adaptation takes place, parameters for time $t+1$ are set to those from time t in order to build the timelined sets (line 12).

It is worth to note that formulas 4 and 5 involve solving a kind of optimization problem, which is left to the specific mining technique. By this, the interaction between different kinds of resources, quality measures and adaptation end-points is regarded. A significantly more complex approach is to include qualities and resources into *one single* optimization problem. But we expect any such problem to be much too complex in order to be solved in practicable time.

Applying the model as proposed, there is only one question unanswered until now: What quality is provided when querying the mining result of an arbitrary time interval? As stream mining algorithms should support such queries but quality measures differ between different time intervals, the answer to this question is fundamental to support meaningful quality awareness. Thus, we define a

last function

$$\xi : Q_T \times N \times N \rightarrow Q \tag{6}$$

which determines an instance Q of quality measures extracted from the gathered timelined qualities Q_T with respect to a given interval of time.

3.3 Requirements on Algorithms

Though we try to capture most of the existing data mining algorithms on data streams, the proposed framework is not applicable to all mining algorithms. There are some basic requirements that algorithms need to fulfill in order to be extendable using our model.

Naturally, one of the crucial requirements is that parameters must exist in the algorithm, so that we can choose adaption factors. Moreover, a strong correlation between the adaption factors and the algorithm's resource requirements aids precisely estimating the quality of the output. Also, in order to anticipate an algorithm's resource requirements as well as the quality of its output, the algorithm must show homogeneous behavior when provided with an input stream whose properties are maintained homogeneous as well. For example, most threshold based stream mining algorithms meet these requirements.

Another important property is that there should exist a partitioning into independent sections in the mining result of an algorithm. That means that different values of the adaption factors only have "local" effects in the mining results. This also indicates that it must be possible to query each of these independent sections separately, since otherwise the lowest quality setting of the adaption factors in the history of the data stream processing will determine the quality of the overall mining results. As an example, consider frequent itemset stream mining using a landmark window model, where the error threshold is one of the adaption factors. Since we can only query the complete history of the stream, the lowest value of the error threshold ever used while processing the stream will determine the overall output quality of this algorithm. Note that this last aspect is not a strict requirement but rather a helpful property. If an algorithm does not satisfy this requirement, it will still be applicable for the proposed framework, but it will not be able to "recover" from low quality adaption factor settings.

Currently, we are adopting the resource- and quality-aware mining techniques we proposed in former works to the presented framework. Despite some last formulas, this is almost done, which shows that the introduced formalization is suitable for application to existing and future proposals.

4 Conclusions and Future Work

Mining data streams stresses our computational resources with regard to processing power, memory requirements, energy and communication. In order to

ensure the continuity and consistency of a data stream mining process, adaptation to available resources is required. Although adaptation is crucial for the success of the data mining process, its effect on the quality is of concern. Thus, in this paper we propose a Quality Guaranteed Resource-Aware (QGRA) data stream mining approach. The objective of this approach is two-fold. Firstly, the adaptation is done while maintaining a guaranteed QoS set by the user. Secondly, utilization of resources is achieved through mapping of required resources to quality measures. If the same quality could be achieved with less resources, only the required resources are consumed with appropriate parameter settings.

Our work provides the mathematical foundation to add quality-guaranteed resource awareness to most stream mining algorithms. Concrete instances of the formulas given in this paper must be implemented by the respective algorithm. We are currently working on applying our model to existing algorithms for the three main data mining tasks, clustering, classification, and frequent itemset mining.

The paper presents a pioneering work to address the two most important challenges in data stream mining, namely, resource constraints and quality of the output model. We believe in the flexibility and suitability of the proposed framework in order to cover existing and following approaches as well as meeting the special challenges of stream mining. By this, we built a valuable and essential basis for future work on quality guaranteed resource-aware stream mining.

References

1. Jiang, N., Gruenwald, L.: Research issues in data stream association rule mining. SIGMOD Rec. **35** (2006) 14–19
2. Motwani, R., Widom, J., Arasu, A., Babcock, B., Babu, S., Datar, M., Manku, G.S., Olston, C., Rosenstein, J., Varma, R.: Query processing, approximation, and resource management in a data stream management system. In: CIDR. (2003)
3. Tan, P., Kumar, V.: Interestingness measures for association patterns: A perspective. KDD WS on Postprocessing in Machine Learning and Data Mining (2000)
4. Franke, C., Hartung, M., Karnstedt, M., Sattler, K.: Quality-Aware Mining of Data Streams. In: Information Quality (ICIQ). (2005) 300–315
5. Chi, Y., Wang, H., Yu, P.S.: Loadstar: load shedding in data stream mining. In: VLDB, VLDB Endowment (2005) 1302–1305
6. Gaber, M.M., Yu, P.S.: A framework for resource-aware knowledge discovery in data streams: holistic approach with its application to clustering. In: SAC. (2006) 649–656