14

Mining Frequent Itemsets over Recent Data Stream Based on Genetic Algorithm

Zhou Yong[*], Han Jun and Guo He
School of Software of Dalian University of Technology, Dalian
China

1. Introduction

Data stream is massive sequence of data elements generated at a rapid rate which is characterized by continuously flowing, high arrival rate, unbounded size of data and real-time query requests. The knowledge embedded in a data stream is more likely to be changed as time goes by. Identifying the recent change of a data stream, especially for an online data stream, can provide valuable information for the analysis of the data stream. Frequent patterns on a data stream can provide an important basis for decision making and applications. Because of the data stream's fluidity and continuity, the information of frequent patterns changes with the new data coming.

Mining over data streams is one of the most interesting issues of data mining in recent years. Online mining of data streams is an important technique to handle real-world applications, such as traffic flow management, stock tickers monitoring and analysis, wireless communication management, etc. In most of the data stream applications, users tend to pay more attention to the mode information of the recent data stream. Therefore, mining frequent patterns in recent data stream is a challenging work. The mining process should have one-pass algorithm, high efficiency of updating, limited space cost and online response of queries. However, most of mining algorithms or frequency approximation algorithms over a data stream could not have high efficiency to differentiate the information of recently generated data elements from the obsolete information of old data elements which may be no longer useful or possibly invalid at present.

Many previous studies contributed to efficient mining of the frequent itemsets over the streams. Generally, three processing models are used which are the landmark model, the sliding window model and the damped model[1]. The landmark model analyzes the stream in a particular window, which starts from a fixed timestamp called landmark and ends up with the current timestamp. For the sliding window model case, the mining process is performed over a sliding window of a fixed length. Based on the sliding window model, the oldest data is pruned immediately when a new data arrives. The damped model uses the entire stream to compute the frequency with a decay factor *d*, which makes the recent data more important than the previous ones.

[*] Supported by Fundamental Research Funds for the Central Universities No. DUT10JR15

Mining frequent patterns on a data stream has been studied in many ways and the mining methods include Dstree[2,3,4], FP-tree[5,6,7]as well as estDec[11] algorithm.

FP-Tree structure is generated by reading data from the transaction database. Each tree node contains an item marker and a count. The count shows transaction numbers which is mapped in the path. Initially FP-Tree contains only one root node, marked with the symbol null. First of all it scans the data set to determine the support count of each item to discard non-frequent items, and list the frequent items in descending order according to their support count. Then, it scans data set secondly to construct FP-Tree. After reading the first transaction data, it can create a node and the path of the first transaction and give the transaction a code. We design the frequency count as 1 to all of the nodes on the path. Then, it should read each of the other transaction data in order to form different paths and nodes. The frequency count will be adjusted until each transaction is mapped to a path on FP-Tree. After reading all the transaction formation to construct the FP-Tree, the FP-Stream algorithm could be used on FP-Tree to mine its frequent itemsets.

DStree algorithm is a relatively new algorithm for mining frequent itemsets which have the concept of nested sub windows in sliding window. DStree algorithm separates the current transaction database data into blocks, then statistic frequent itemsets in the current window. When a next block of data comes to the moment, the prior block data becomes the historical data. The second block of data replace the first one. Some of the information are available in current DStree and prepare for the next generation of a DStree

estDec algorithm is a effective way to mine frequent itemsets of current on-line data stream. Each node of estDec algorithm model tree contains a triple (*count, error, Id*). For the relevant item *e*, its number is shown by *count*. The maximum error count of *e* is shown with *error* and *Id* is the determined factor of *e* wich contains the most recent transactions. estDec algorithm is divided into four parts: update parameter, update count, the delay difference and choose frequent items.

As using model tree in FP-Tree , DStree and estDec algorithm, it is difficult to make the algorithm computing parallel and the algorithm run time is also difficult to reduce.

With the development of the card, GPU (Graphic Process Unit) become more and more powerful. It has transcended the CPU computation not only on graphic but also on scientific computing. CUDA is a parallel computation framework which is introduced by NVIDIA. The schema makes GPU be able to solve complex calculations. It contains the schema CUDA instruction set and internal computation engine. GPU is characterized by processing parallel computation and dense data, so CUDA suites large-scale parallel computation field very well[12].

This work proposes a NSWGA (Nested Sliding Window Genetic Algorithm) algorithm. Firstly, NSWGA gets the current data stream through the sliding window and uses a nested sub-window dividing up the data stream in current window into sub-blocks; then, the parallel idea of genetic algorithm and parallel computation ability of GPU are used to seek frequent itemsets in the nested sub-window; at last, NSWGA gets the frequent patterns in the current window through the frequent patterns of the nested sub-windows.

This chapter is organized as follows. Theoretical foundation is described in Section 2. The algorithm is designed for Nested Sliding Window Genetic Algorithm of mining frequent itemsets in data streams in Section3. In Section 4, comprehensive experiments for the algorithm are implemented in built environment and give the comparison with other methods. Moreover, algorithm analysis is also proposed for mining time-sensitive sliding windows in this section.. Finally, we summarize the work in Section 5.

2. Theoretical foundation

The study combines the sliding window techniques, frequent itemsets, genetic algorithm and parallel processing technology.

Sliding window has been used in the network communication, time-series data mining, data stream mining and so on. This algorithm uses the sliding window [9,10] to obtain the current data stream.

Definition 1 sliding window: For a positive number ω1, a certain time T, data sets D = (d_0, d_1 ,..., d_n) fall into the window SW(the size of window SW is ω1), the window SW is called the sliding window.

Definition 2 nested sub-window: For a positive number ω2, a certain time T, the newest data set dn in sliding window SW falls into the nested window NSW (the size of NSW is ω2), the nested window NSW is called the nested sub-window.

As shown in Figure 1, the application of sliding window for dynamic updating of data sets is explained.

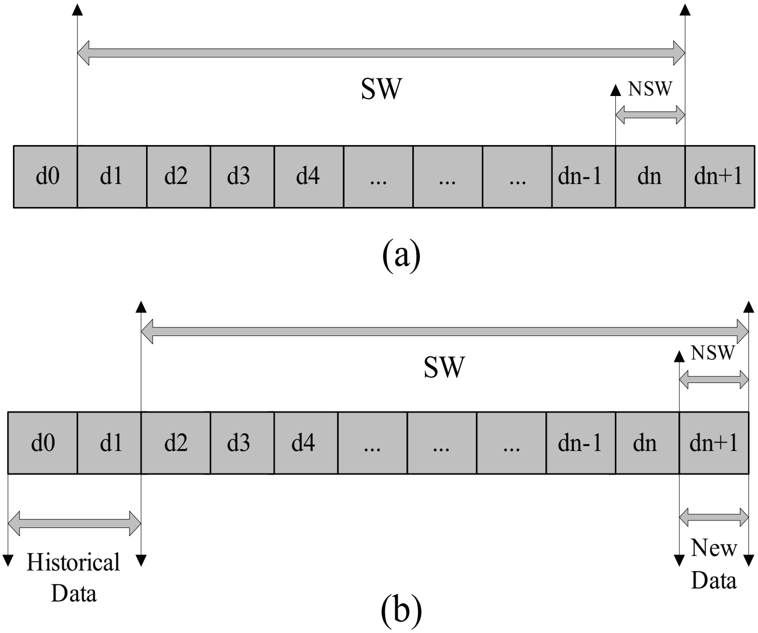


Fig. 1. Dynamic updating of the data in sliding window

Definition 3 frequent itemsets in sliding window: For the current data in sliding window, a collection of items I = {i_1, i_2, ..., i_n} , transaction iterm data set S = {s_1, s_2 ,..., s_n}, each transaction iterm is a collection of items, s⊆I ◦ If X⊆S, then X is an itemset. If there are k elements in X, we call X the k-itemsets. With respect to an itemset X, if its support degree is greater than or equal to the minimum support threshold given by the user, then X is called the frequent itemsets.

Genetic algorithm starts the search process from an initial population. Each individual in the population is a possible frequent pattern. We use the genetic algorithm to achieve the result mainly through crossover, mutation and selection [8]. After several generations of selection, we achieve a final frequent itemsets. The major rules and operators in genetic algorithm are as follows:

1. Coding rule: this work codes with the integer. For example, each transaction item has ABCDE five attributes in a data stream, the transaction item which is coded 21530 expresses that we take the second value of A attribute, take the first value of B attribute, and analogizes in turn, we use 0 to express that we do not consider the value of E attribute.
2. The fitness function: F_i=W_i/W_Z, F_i is the support degree of transaction item i, W_i is the number of the transaction items which have the same value for each attribute, W_Z is the total number of transaction items in the window.
3. The selection operator: This algorithm uses the Roulette Wheel Selection. For individual i, its fitness degree F_i, the population size M, then its probability of being selected is expressed as $p_i = F_i / \sum_{i=1}^{M} F_i$, (i=1, 2, … , M).
4. Crossover: This algorithm uses One Point Crossover. If the parent chromosomes are A ($a_1a_2a_3$ ... a_i ... a_n) and B ($b_1b_2b_3$ ... b_i ... b_n), after cross operation, the daughter chromosomes are A_1 ($a_1a_2a_3$ ... b_i ... b_n) and B_1 ($b_1b_2b_3$ ... a_i ... a_n).
   Crossover operator is mainly used to interchange some genes between the parent chromosomes. Through the operation between two individuals of parent generation, we get the daughter generation. Thus, daughter generation would inherit the effective models of the parent generation.
5. Mutation Operator: The algorithm uses the Simple Mutation. If the parent chromosome is A ($a_1a_2a_3$ ... a_i ... a_n), after the variation, the daughter chromosome becomes A_1 ($a_1a_2a_3$ ... b_i ... a_n).

Mutation operation changes some genes randomly to generate new individuals. Mutation operation is an important cause to obtain global optimization. It helps to increase the population diversity, but in this algorithm, the corresponding genes which are required to generate the frequent itemsets already exist, so we use a lower mutation rate.

When we establish the parallel part in the program, we can let this part run into GPU. The function which runs in GPU is called kernel (kernel function). A kernel function is not a complete program, but the parallel part of the entire CUDA program[13,14]. A complete CUDA program execution is shown in figure 2. The graph shows that in a kernel function there are two parallel levels, the parallel blocks in the grid and the parallel threads in the block.

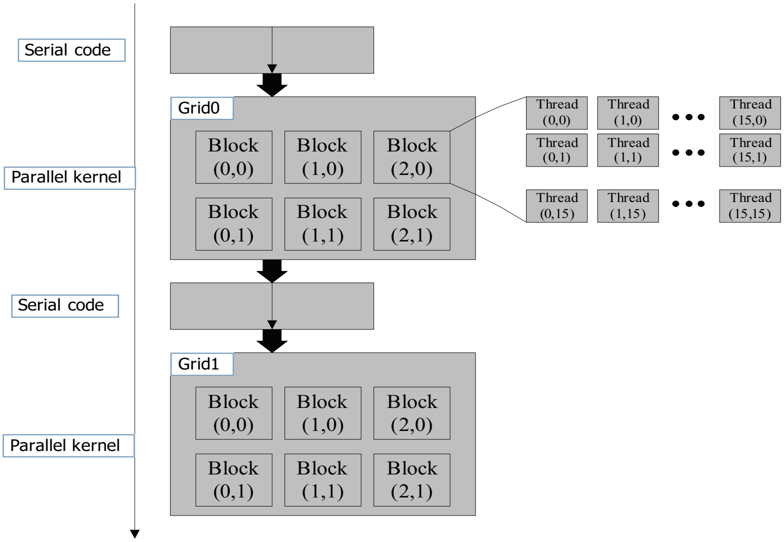


Fig. 2. CUDA programme model

3. Algorithm design

NSWGA uses the sliding window to get the recent data and uses genetic algorithms to mine frequent itemsets of the data in the current window.

3.1 Algorithm description

Input data streams to be mined

Output frequent itemsets of recent data stream

NSWGA algorithm is divided into three parts: (1) NSWGA uses the parallelism of genetic algorithm to search for the frequent itemsets of the latest data in the nested sub-window. (2)The final frequent itemsets of the sliding window are obtained by the integrated treatment of this series of frequent itemsets in nested sub-windows. (3)With the new data coming, the expired data is deleted periodically. Repeat the above two operations.

In the first part, the current frequent itemsets in NSW is obtained. The process is shown as figure 3.

Step 1. Set the size of sliding window SW ω1. Set the size of nested sub-window NSW ω2.Window sizes are determined by the properties of the data stream. ω1 depends on how many current affairs whose frequent itemsets we are interested in. ω2 depends on the processing capability of the algorithm and our statistical frequency.
Given the support threshold S, fitness function $F_i=W_i/W_Z$, when $F_i \geq S$, transaction iterm i is a frequent pattern of the data set in sliding window.
The iteration times T depends on the number of attributes that a transaction iterm includs and the scope of the attribute values and the original population size. The

role of nested sub-window is to avoid repeatedly processing the data which is still in the sliding window after the old data out of the sliding window.

Let the crossover probability is P, the individual mutation probability is Q. To implement parallel computing, the data in the nested sub-window is divided into Z segments.

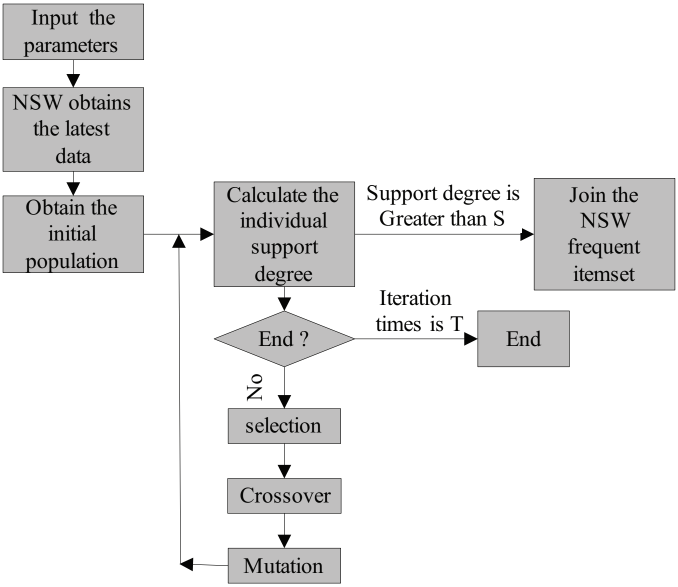

Input the parameters
NSW obtains the latest data
Obtain the initial population
Calculate the individual support degree
Support degree is Greater than S
Join the NSW frequent itemset
End ?
Iteration times is T
End
No
selection
Crossover
Mutation

Fig. 3. The generation of initial population.

Step 2. Use the nested sub-window to achieve the latest data, get frequent 1-itemsets of the data, encode the frequent 1 - itemsets to integer strings, and combine the frequent 1 - itemsets randomly to constitute the initial population in the nested sub-window. The individuals of this population are possible frequent patterns.

1 Statistics the number of I_1, I_2, I_3 in A attribute;
2 Statistics the number of I_1, I_2, I_3 in B attribute;
3 Statistics the number of I_1, I_2, I_3 in C attribute;
4 Reserve the value which is greater than or equal to the threshold S, let others are 0 (in this case, S takes 3);
5 Remove the all zero -line, set non- zero values according to their original row;
6 Line up every non-zero value and keep its original location in the line, fill in the rest position with 0;
7 Combine non-zero iterms according to their original location.Constitute the initial population with frequent 1 –itemsets and the combination iterms.

The process is shown in Figure 4.

Step 3. Calculating the individual fitness degree is the process that individuals in the initial population match with the actual transaction iterms. In order to realize parallel matching, we divide the data into Z sections. Although this operation increases the memory expenses, it reduces the running time. It is important in mining frequent patterns of data stream. Make Roulette Wheel Selection according to the fitness degree. Make crossover with the Crossover probability P. Carry on the variation with the variation probability Q. Ascertain the individual fitness degree after scanning the data. Join the individual which satisfies the condition into the frequent itemsets.

Relying on the powerful parallel computing capability of GPU, parallel matching with Z sections, that will reduce a lot of running time, the process is shown in Figure 5.

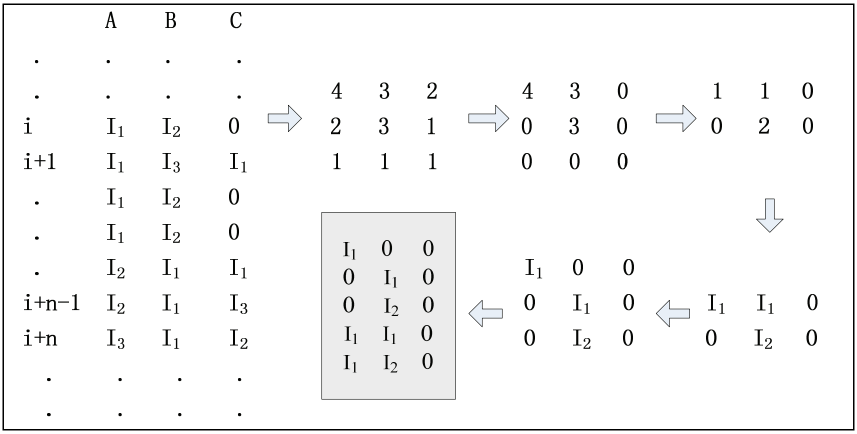


Fig. 4. The generation of initial population

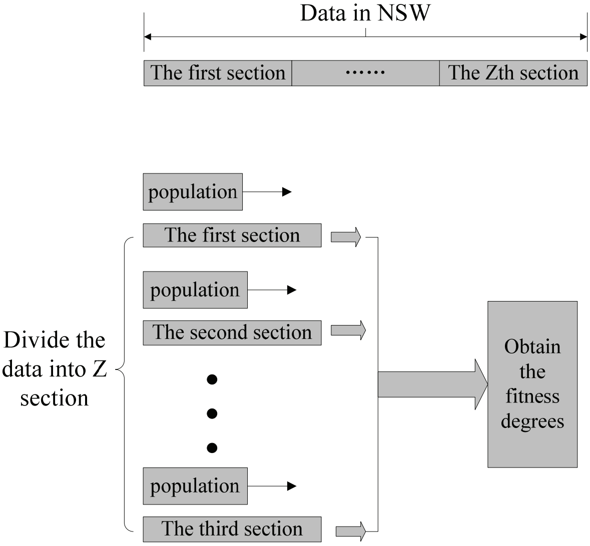


Fig. 5. Parallel computing fitness degree

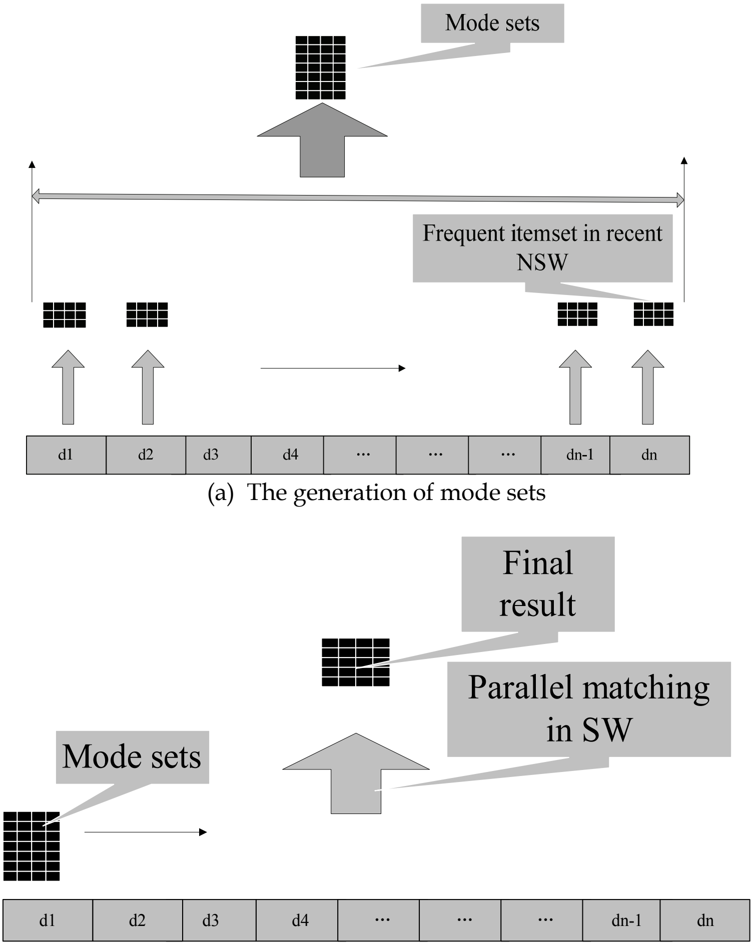
Step 4. If the number of iterative times is smaller than T, the algorithm jumps to the step 3. After T times of iterative computation, finish iterative and obtain the frequent itemsets in current nested sub-window;

In the second part, the final frequent itemsets in sliding window is obtained. The process is shown as step5.

Step 5. Constitute the mode sets with the frequent itemsets that we obtained this time and the previous frequent itemsets obtained in the last M (M = ω1/ω2-1) times. Carry on a search to determine the final frequent itemsets in the sliding window.

```
For i = 1: M+1
    Constitute the mode sets;
End
Make a parallel search in the sliding window SW;
When a mode's support degree is greater than or equal to S, identify it as a final frequent mode;
```

The process is shown in Figure 6 (a) (b).



(a) The generation of mode sets

(b) The generation of final frequent patterns

Fig. 6. The process of obtaining frequent patterns

In the third part, repeat the above two operations dynamically. The process is shown as step6.

Step 6. With the data stream flowing, this algorithm continues to deal with the new incoming data and discard the old data, transfer to step 2 and continue the above operations until the data stream coming to the end.

3.2 NSWGA algorithm analysis

Comparing with other algorithms which use pattern tree to maintain the historical information of data stream, NSWGA processes a quantity of data parallelly at one time, while the pattern tree algorithms process a single transaction item at one time, each transaction item needs match repeatedly. Mining the frequent itemsets of the data in the current window, the time of whole process is not only dependent on the times of scanning the data in the window, but also dependent on the internal basic operation - the number of matching.

Suppose a data stream has N transaction items, each transaction item has V attributes; each attribute has K possible values. The pattern tree algorithms may have K^V frequent pattern search paths. Let the window size is N. When the entire data stream in the window flow over, the necessary calculated amount to get frequent itemsets is N * K * V.

For fp-tree algorithm, when the fp-tree has L paths, the calculated amount is 2 * N * V + V * L, the number L will increase with the threshold of support degree reducing.

When the support degree is S, iteration times of genetic algorithms is T, the number of parallel computing is Z (Z according to the amount of data, in this case set Z 200),the sliding window size is N, the necessary calculated amount to get frequent itemsets is P = P1 + P2 + P3. Thereinto:

P1 = N * V   the calculated amount to get 1 - frequent itemsets;

P2 = V * T * N / S * Z   the calculated amount to get the frequent itermsets in the nested sub-window;

P3= α *V*N/S*Z*M (1<= α <=1/S)   the calculated amount to get the final frequent itemsets.

When the property value K is large, this algorithm has obvious advantage in time complexity. When the number of Z is larger, the runtime will become shorter.

4. Experiment and analysis

4.1 Experiment

In this experiment, we use artificial data sets and the MATLAB and CUDA C language to implement NSWGA algorithm. We use the computer with 2.61GHZ CPU, 2GMB memory, Nvidia GPU C1060, windows XP operating system to test the performance of the algorithm.

The size of the sliding window is 100k. The size of the data set is 200K.With the data flowing, we make statistic every 10K of the data.

1. The analog data stream has three attributes. Each attribute has 10 possible values. The running results of the algorithms are shown in Table 1.

algorithm	suport degree	average runtime
fp-tree	10%	0.156
fp-tree	20%	0.087
fp-tree	30%	0.029
NSWGA	10%	0.087
NSWGA	20%	0.032
NSWGA	30%	0.015

Table 1. The comparison of fp-tree **algorithm** and NSWGA algorithm

2. The analog data stream is the same as above. The running results of the algorithms are shown in Table 2.

algorithm	suport degree	average runtime
Dstree	10%	0.138
Dstree	20%	0.139
Dstree	30%	0.141
NSWGA	10%	0.087
NSWGA	20%	0.032
NSWGA	30%	0.015

Table 2. The comparison 1 of Dstree algorithm and NSWGA algorithm

3. The analog data stream has three attributes. Each attribute has 20 possible values. The running results of the algorithms are shown in Table 3.

algorithm	suport degree	average runtime
Dstree	10%	0.406
Dstree	20%	0.397
Dstree	30%	0.402
NSWGA	10%	0.090
NSWGA	20%	0.041
NSWGA	30%	0.017

Table 3. The comparison 2 of Dstree algorithm and NSWGA algorithm

4.2 Analysis of the experimental results

As shown in Table 1, with the support degree increasing, the frequent patterns of these two algorithms are rapidly reducing, the number of matching is reduced and eventually the runtime will be reduced. However, fp-tree algorithm not only needs to maintain the global frequent pattern tree, but also requires additional time to build a sub-pattern tree for each data segment. Then this algorithm saves the information of the sub-pattern tree to the global frequent pattern tree. With the times of process increasing，the runtime of fp-tree algorithm is becoming longer than NSWGA.

Table 2 shows that, with the support degree increasing, the algorithms which use pattern tree to maintain the information of the frequent patterns such as Dstree algorithm can not reduce the runtime, but NSWGA algorithm is able to save a lot of runtime.

In Table 2, the attribute of analog data has 10 possible property values, and in Table 3 there are 20. With the number of possible property values increasing, the runtime of Dstree algorithm will be greatly increased, while the runtime of NSWGA algorithm almost has no change.

5. Summary

It is important for prediction and decision-making to find frequent items among huge data stream. This chapter presents an approach, namely NSWGA (Nested Sliding Window Genetic Algorithm), about mining frequent itemsets on data stream within the current window. NSWGA uses the parallelism of genetic algorithm to search for the frequent itemset of the latest data in the nested sub-window. The final frequent itemsets of the sliding window is obtained by the integrated treatment of this series of frequent itemsets in nested sub-window. NSWGA captures the latest frequent itemsets accurately and timely on data stream. At the same time the expired data is deleted periodically. As the use of nested windows and the parallel processing capability of genetic algorithm, this method reduced the time complexity.

In this chapter, an algorithm about mining frequent patterns of data stream- NSWGA algorithm is proposed. The main contributions of this algorithm: (1) The parallelism of genetic algorithm is used to mine the frequent patterns of data stream , which reduces the runtime; (2) The algorithm combines the sliding window with genetic algorithm to propose an improved method to obtain initial population; (3) This algorithm gurantees the speed of implementation and query precision.

6. References

[1] Lichao Guon, HongyeSu,YuQu. Approximate mining of global closed frequent itemsets over data streams. Journal of the Franklin Institute 348 (2011) 1052–1081.

[2] Chao-Wei Li, Kuen-Fang Jea. An adaptive approximation method to discover frequent itemsets over sliding-window-based data streams. Expert Systems with Applications 38 (2011) 13386 – 13404.

[3] Syed Khairuzzaman Tanbeer, Chowdhury Farhan Ahmed, Byeong-Soo Jeong *, Young-Koo Lee. Sliding window-based frequent pattern mining over data streams. Information Sciences 179 (2009) 3843 – 3865.

[4] Carson Kai-Sang Leung Quamrul I. Khan.DSTree: A Tree Structure for the Mining of Frequent Sets from Data Streams[C].Hong Kong: Proceedings of the Sixth International Conference on Data Mining. (2006)928–932.

[5] Tzung Pei Hong, Chun Wei Lin, Yu Lung Wu. Maintenance of fast updated frequent pattern trees for record deletion. Computational Statistics & Data Analysis,Vol.53,(2009)2485-2499.

[6] Han J, Jian P.Miningfrequent patterns without candidate generation[C].Dallas,TX: Proceedings of ACM SIGMOD International Conference on Management of Data, (2000)1-12.

[7] Zhi-Xin Feng, Zhong Cheng. An algorithm for mining maximal frequent patterns based on FP-tree. Computer Engineering, Vol.30, (2004) 123-124.

[8] Wang Xiaoping, Cao Liming. Genetic algorithm - theory, application and software implementation [M]. Xi'an：Xi'an Jiaotong University Press,(2002).

[9] Syed Khairuzzaman Tanbeer, Chowdhury Farhan Ahmed, Byeong-Soo Jeong. Sliding window-based frequent pattern mining over data streams. Information Sciences: an International Journal, Vol.179,(2009) 3843 - 3865.

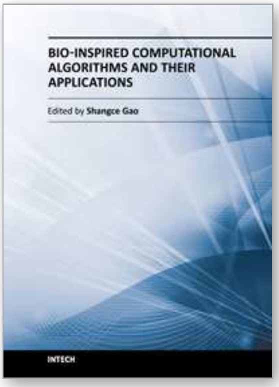
[10] Joong Hyuk Chang, Won Suk Lee. A sliding window method for finding recently frequent itemsets over online data streams. Journal of Information Science and Engineering ,Vol.20,(2004)753 - 762.

[11] Joong Hyuk Chang, Won Suk Lee.Finding Recent Frequent Itemsets Adaptively over Online Data Streams[C]. Washington: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining.(2003) 487 – 492.

[12] F.Molnar Jr.,T.Szakaly,R.Meszaros,I.Lagzi..Air pollution modelling using a Graphics Processing Unit with CUDA[J]. Computer Physics Communications, 2010,181(1):105-112.

[13] NVIDIA Corporation.CUDA programming guide[Z].2008.

[14] Harish P, Narayanan PJ. Accelerating large graph algorithms on the GPU using CUDA[C].Springer Heidelberg,2007:367-390.



Bio-Inspired Computational Algorithms and Their Applications

Edited by Dr. Shangce Gao

ISBN 978-953-51-0214-4
Hard cover, 420 pages
Publisher InTech
Published online 07, March, 2012
Published in print edition March, 2012

Bio-inspired computational algorithms are always hot research topics in artificial intelligence communities. Biology is a bewildering source of inspiration for the design of intelligent artifacts that are capable of efficient and autonomous operation in unknown and changing environments. It is difficult to resist the fascination of creating artifacts that display elements of lifelike intelligence, thus needing techniques for control, optimization, prediction, security, design, and so on. Bio-Inspired Computational Algorithms and Their Applications is a compendium that addresses this need. It integrates contrasting techniques of genetic algorithms, artificial immune systems, particle swarm optimization, and hybrid models to solve many real-world problems. The works presented in this book give insights into the creation of innovative improvements over algorithm performance, potential applications on various practical tasks, and combination of different techniques. The book provides a reference to researchers, practitioners, and students in both artificial intelligence and engineering communities, forming a foundation for the development of the field.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Zhou Yong, Han Jun and Guo He (2012). Mining Frequent Itemsets over Recent Data Stream Based on Genetic Algorithm, Bio-Inspired Computational Algorithms and Their Applications, Dr. Shangce Gao (Ed.), ISBN: 978-953-51-0214-4, InTech, Available from: http://www.intechopen.com/books/bio-inspired-computational-algorithms-and-their-applications/mining-frequent-itemsets-over-recent-data-stream-based-on-genetic-algorithm

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821