Algorithms **2013**, *6*, 747-761; doi:10.3390/a6040747

OPEN ACCESS

algorithms

ISSN 1999-4893

www.mdpi.com/journal/algorithms

Article

Multi-Core Parallel Gradual Pattern Mining Based on Multi-Precision Fuzzy Orderings

Nicolas Sicard [1], Yogi Satrya Aryadinata [2], Federico Del Razo Lopez [3], Anne Laurent [2,*] and Perfecto Malaquias Quintero Flores [4]

[1] Efrei-AllianSTIC, Villejuif 94800, France; Email: nicolas.sicard@efrei.fr

[2] LIRMM, University Montpellier 2 - CNRS, Montpellier 34095, France; E-Mail: Yogi.Aryadinata@lirmm.fr

[3] Toluca Institute of Technology, Toluca 64849, Mexico; E-Mail: delrazo@ittoluca.edu.mx

[4] Apizaco Institute of Technology, Apizaco 90300, Mexico; E-Mail: malakaz@prodigy.net.mx

* Author to whom correspondence should be addressed; E-Mail: laurent@lirmm.fr; Tel.: +33(0)786982294.

Received: 17 September 2013; in revised form: 16 October 2013 / Accepted: 21 October 2013 / Published: 1 November 2013

Abstract: Gradual patterns aim at describing co-variations of data such as *the higher the size, the higher the weight*. In recent years, such patterns have been studied more and more from the data mining point of view. The extraction of such patterns relies on efficient and smart orderings that can be built among data, for instance, when ordering the data with respect to the size, then the data are also ordered with respect to the weight. However, in many application domains, it is hardly possible to consider that data values are crisply ordered. When considering gene expression, it is not true from the biological point of view that Gene 1 is more expressed than Gene 2, if the levels of expression only differ from the tenth decimal. We thus consider fuzzy orderings and *fuzzy gamma rank correlation*. In this paper, we address two major problems related to this framework: (i) the high memory consumption and (ii) the precision, representation and efficient storage of the fuzzy concordance degrees versus the loss or gain of computing power. For this purpose, we consider multi-precision matrices represented using sparse matrices coupled with parallel algorithms. Experimental results show the interest of our proposal.

Keywords: parallel data mining; gradual patterns; fuzzy rank

---

1. Introduction

In data mining, mining for frequent patterns (In this paper, the words *item* and *pattern* are considered as being synonyms.) has been extensively studied during recent years. Among the patterns that can be discovered, gradual patterns aim at describing co-variations of attributes, such as *the higher the size*, *the higher the weight*. Such a gradual pattern relies on the fact that when the age increases, the salary also increases, people being ranked regarding their age and salary. However, real world databases may contain information that can hardly be ranked in a crisp manner. For instance, gene expression levels are measured by instruments and are imperfect. For this reason, an expression level can hardly be declared as being greater than another one if they only differ from a small value. We thus claim that orderings must be considered as being soft. Fuzzy orderings and fuzzy ranking indeed allow to handle vagueness, ambiguity or imprecision present in problems for deciding between fuzzy alternatives and uncertain data [1–3,12,16]. However, though there are great benefits to fuzzy orderings and fuzzy rank correlation measures, these techniques prevent us from considering binary relations (greater than / lower than) and binary representations in machine which are efficient from the memory consumption and computation time (binary masks) points of view. The representation and efficient storage of the vagueness and imprecision of the data is indeed a complex challenge as studied in [2]. We thus propose a framework to address the high memory consumption, the representation, precision and efficient storage of the fuzzy concordance degrees, by using sparse matrices and high performance computing (parallel programming).

This paper is organized as follows: Section 2 reports existing work on fuzzy orderings, gradual pattern mining and parallel data mining. Section 3 presents our gradual item set mining algorithm and our framework to address the high memory consumption, the representation, precision and efficient storage of the fuzzy concordance degrees. Experimental results are presented in Section 4. Section 5 is our conclusion.

2. Related Work

In this section, we recall the definition of gradual pattern in the particular context of fuzzy orderings (Section 2.1) before presenting parallel data mining (Section 2.2).

2.1. Gradual Pattern Mining and Fuzzy Orderings

Frequent gradual patterns are patterns like *the older, the higher the salary*. They are extracted from databases whose schema is defined over several attributes, also called items, which domains must be equipped with a total ordering. In our framework, we consider the following definitions of *gradual item*, *gradual itemset*, *concordant couple* and *support* of a *gradual itemset*.

Let db be a database constituted of n data records (objects) denoted by $O = \{o_1, o_2, ..., o_n\}$ defined over the database schema of m attributes $A = \{A_1, A_2, ..., A_m\}$ which domains are equipped with an order relation. The set of objects O must also be equipped with an order relation so that db is said to be a gradual database.

Table 1 reports an example of such a database where there are three attributes related to three characteristics of five fruits. These attributes are numeric and the order relation is the "*is lower than*"one over every single attribute. Regarding a set of attributes A, objects are ordered by considering that an object precedes (resp. succeeds) another one if its value is lower than (resp. greater than) the value of the second object on every attribute from A.

Table 1. Example of a database of some fruit characteristics: size, weight, and sugar rate.

Id	A_1:Size	A_2:Weight	A_3:Sugar Rate
o_0	6	6	5.3
o_1	10	12	5.1
o_2	14	4	4.9
o_3	23	10	4.9
o_4	6	8	5.0
o_5	14	9	4.9

A *gradual item* is defined as a pair $(A_l \in db, v)$ where v is a variation that can be ascending ($\uparrow$) if the attribute values increase, or descending ($\downarrow$) if the attribute values decrease, *i.e.*, $\{A_l\uparrow\} \simeq \{A_l(o_i)<A_l(o_j)\}$ and $\{A_l\downarrow\} \simeq \{A_l(o_i)>A_l(o_j)\}$ for *i=1, 2, ..., n*, for *j=i+1, ..., n*, $i \neq j$ and $l \in \{1,2, ..., k\}$ [4,5].

A *GI* (gradual itemset) is a combination of *gradual items* of the form $GI=\{A_1\downarrow A_2\downarrow A_3\uparrow\}$ interpreted as $\{$*The lower* A_1*, the lower* A_2*, the higher* $A_3\}$. Where the size (k) of a *GI* is defined as the number of *gradual items* contained in the *GI*, such that $k \in \{2, 3, 4,..., m\}$, each *gradual items* $\in$ *GI* is unique [4,5].

For instance, *(size,* $\uparrow$*)* is a gradual item and $\{$*(size,* $\uparrow$*), (weight,* $\uparrow$*)*$\}$ is a gradual itemset. A *GI* is an interesting pattern if *support(GI)* is greater than or equal to the user-predefined minimal support called *minimum threshold* (minsup).

Several definitions have been proposed in order to compute the support of a gradual pattern within a database. In our approach, we opted for the framework of the gradual dependency interpretation framework based on *induced rankings correlation* and *concordant couple* concept [2,4,13,15]. A *concordant couple (cc)* is an index pair, where the records (o_i, o_j) satisfy all the variations v expressed by the involved *gradual items* in a given *GI* of size k, e.g., let $GI = \{(A_1, \downarrow), (A_2, \downarrow)\}$ with size $k = 2$, an index pair *cc(i, j)* is a *concordant couple* if $((A_1(o_i)>A_1(o_j)\ implies\ A_2(o_i)>A_2(o_j))$, where $i=(A_1(o_i),A_2(o_i))$ and $j=(A_1(o_j),A_2(o_j))$ [2,4].

In this framework, the *support* of a *GI* is computed as:

$$support(GI) = \frac{\sum_{i=1}^{n} \sum_{j \neq i} cc\,(i,j)}{n(n-1)} \quad (1)$$

Given a gradual pattern and a database, it can be represented using a binary matrix which helps computing the support (whatever the support computation technique), as presented in [6] and on Fig 2.

In [5,7], we have shown that fuzzy orderings must be considered in order to better represent the real world where data cannot always be crisply ordered. In this context, we consider a framework based

on the principles of Kendall's tau (rank correlation coefficient), Goodman's and Kruskal's gamma rank correlation measure, Bodenhofer's and Klawonn's fuzzy gamma rank correlation measure denoted by $\tilde{cp}(i,j)$ ranging from 0 to 1. As the degree is no more binary, we have to consider an extension of the matrices, as described in [7]. Figure 2 illustrates the structure of the matrix of fuzzy concordance degrees $\tilde{cp}(i,j) \in [0,1]$, represented with a precision of 2 and 3 bits. In order to address the problem of the precision of the representation and efficient storage of each concordance degree $\tilde{cp}(i,j)$, we consider the storage requirements for the binary case and fuzzy case.

Figure 1. Illustration of the binary matrix of concordant couple and computing of support.

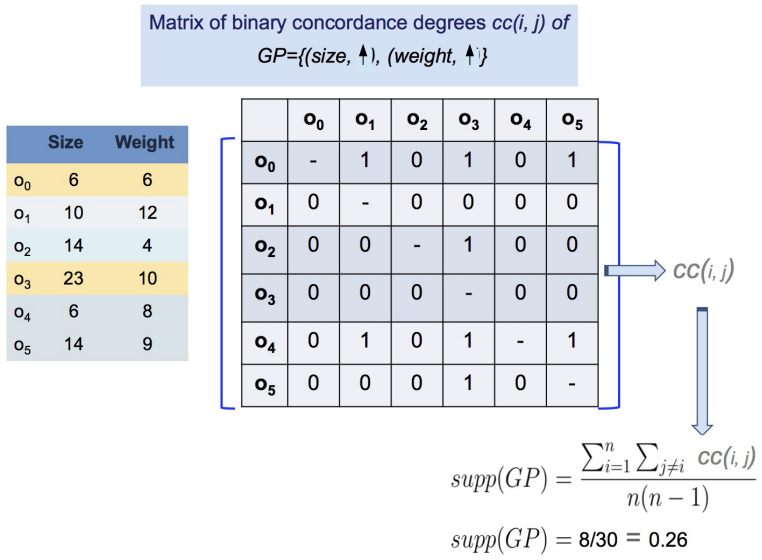
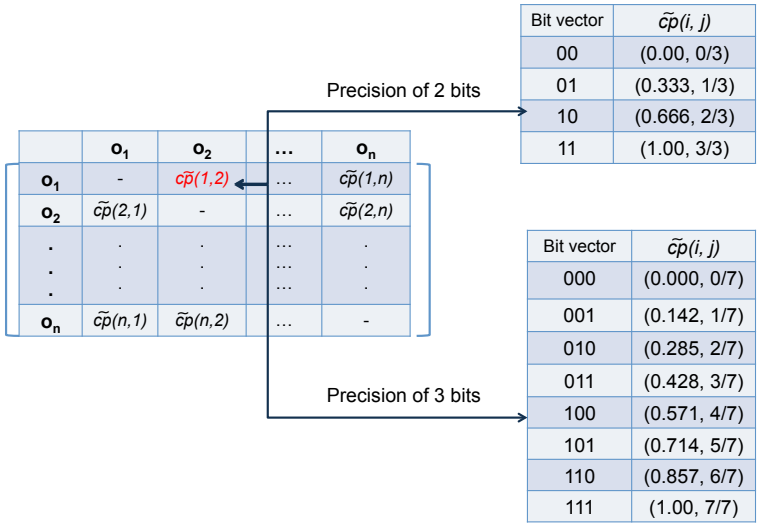


Figure 2. Illustration of the matrix of fuzzy concordance degrees represented with a precision of 2 and 3 bits.



We use a precision of one bit for the binary case, *i.e.*, for each $\tilde{cp}(i,j) \in \{0, 1\}$, as it is sufficient to represent and store a $\{0|1\}$. In the proposed multi-precision matrices, each value $mpm(i,j)$ is represented by a bit-field integer value containing up to 52 bits. Double precision floating point values

of fuzzy concordance degrees $\tilde{cp}(i,j)$ are calculated through the ratio $fl(mpm(i,j))/fl(mmax)$, where $fl(x)$ is the floating point real representation of the integer value x and $mmax$ is the maximum integer value representable in $mpm(i,j)$. For instance, the corresponding fuzzy concordance degree of the 010 (binary) integer value will be $fl(010)/fl(111) = 2.0/7.0 = 0.428$ (Figure 2).

In [7], we considered fuzzy orderings and the management of the fuzzy degrees using Yale matrices. However, in [7], the fuzzy degrees are represented using floating numbers which are memory consuming. In this paper, we thus consider multi-precision representations and their efficient implementation using integer values represented as variable length bit-fields.

2.2. Parallel Data Mining

Parallel computing has recently received lots of interest for use in data mining. New generations of multicore processors and GPUs provide ways to exploit parallelism to reduce execution time [14]. This will allow larger (Big Data) problems to be worked on.

The parallel programming models are roughly divided into three categories:

- Distributed memory systems (each processor has its own system memory that cannot be accessed by other processors, the shared data are transferred usually by message passing, e.g., sockets or message passing interface (MPI));
- Shared memory systems where processors share the global memory, they have direct access to the entire set of data. Here, accessing the same data simultaneously from different instruction streams requires synchronization and sequential memory operations;
- Hierarchical systems (a combination of shared and distributed models, composed by multiprocessor nodes in which memory is shared by intra-node processors and distributed over inter-node processors).

In multicore architectures, a parallel program is executed by the processors through one or multiple control flows referred to as processes or threads [8,9].

3. Parallel Fuzzy Gradual Pattern Mining Based on Multi-Precision Fuzzy Orderings

In this section, we detail our approach.

3.1. Managing Multi-Precision

Concerning the implementation of the matrices of concordance degrees $\tilde{cp}(i,j)$, we address two important issues: (i) the memory consumption; and (ii) the precision of the representation of the concordance degrees of each $\tilde{cp}(i,j)$.

In order to reduce memory consumption, we represent and store each matrix of concordance degrees according to the *Binary Fuzzy Matrix Multi-precision Format*, where each $\tilde{cc}(i,j) \in [0, 1]$ is represented with a precision of 2, 3, or more up to 52 bits.

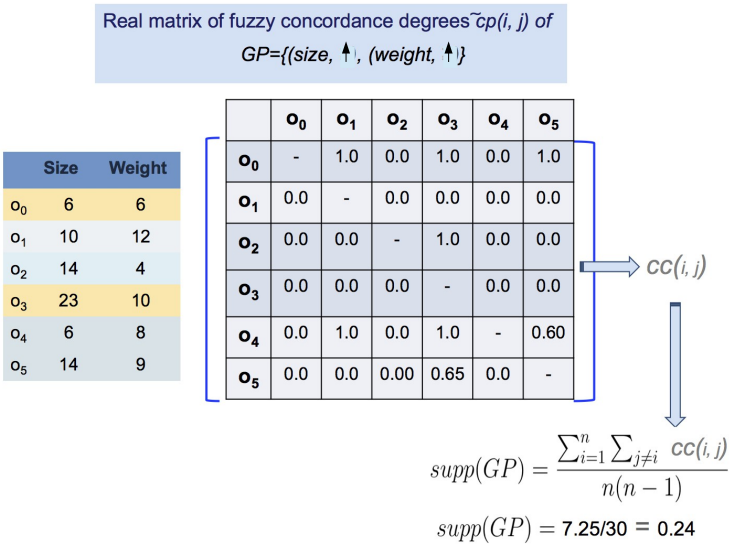
Because we generate itemset candidates from the frequent k-itemsets, only matrices of the $(k-1)$-level frequent gradual itemsets are kept in memory while being used to generate the matrices of the

(k)-level gradual itemset candidates. If the *support* of a *gradual itemset* ($\mathcal{C}_{k,q}$) is less than *minimum threshold*, then the $\mathcal{C}_{k,q}$ is pruned and its matrix of fuzzy concordance degrees $\tilde{cp}(i,j)$ is removed.

As seen in the previous section, fuzzy orderings are interesting but consume large memory slots when they are stored as floating numbers.

On the other hand, binary matrices are very efficient regarding both memory and time consumption, we thus consider binary vectors in order to represent the fuzzy degrees. The size of these vectors determine the precision we manage. Figure 3 shows how values are represented at the 3 bits precision.

Figure 3. Illustration of the real matrix of fuzzy concordance degrees.

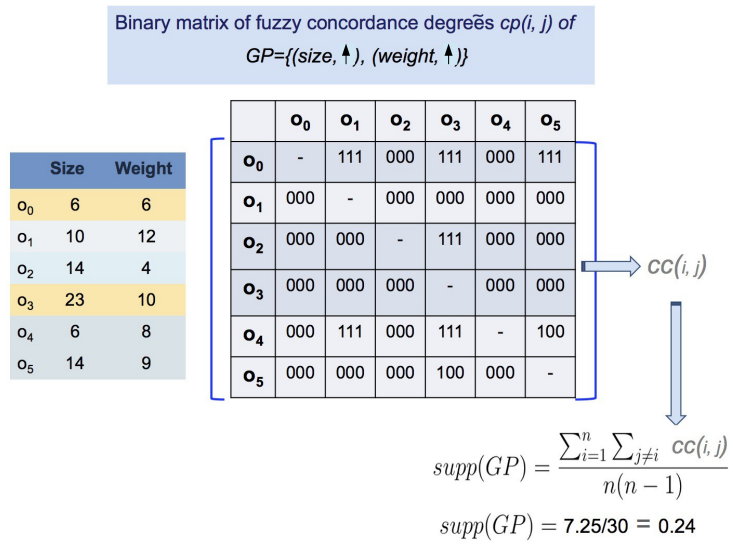


	Size	Weight
o_0	6	6
o_1	10	12
o_2	14	4
o_3	23	10
o_4	6	8
o_5	14	9

	o_0	o_1	o_2	o_3	o_4	o_5
o_0	-	1.0	0.0	1.0	0.0	1.0
o_1	0.0	-	0.0	0.0	0.0	0.0
o_2	0.0	0.0	-	1.0	0.0	0.0
o_3	0.0	0.0	0.0	-	0.0	0.0
o_4	0.0	1.0	0.0	1.0	-	0.60
o_5	0.0	0.0	0.00	0.65	0.0	-

Each $\tilde{cp}(i,j) \in [0,1]$ is thus represented with a precision ranging from 1 bit (crisp case) to n bits (52 in our implementation). n bits allow to represent up to 2^n values. Figure 4 shows the real matrix of fuzzy concordance degrees. Figure 2 shows how to represent values at precision of 3 bits.

Figure 4. Illustration of the binary matrix of fuzzy concordance degrees with a precision of three bits.



	Size	Weight
o_0	6	6
o_1	10	12
o_2	14	4
o_3	23	10
o_4	6	8
o_5	14	9

	o_0	o_1	o_2	o_3	o_4	o_5
o_0	-	111	000	111	000	111
o_1	000	-	000	000	000	000
o_2	000	000	-	111	000	000
o_3	000	000	000	-	000	000
o_4	000	111	000	111	-	100
o_5	000	000	000	100	000	-

In our Algorithm 1, the concept of matrix concordant degrees plays an important role.

Algorithm 1 Fuzzy Orderings-based Gradual Itemset Mining

Data: db (Database), # m (Attributes), # n (Records), $minimum\_threshold$.
Output: Frequent Gradual Itemsets $\mathcal{F}_k$
$\mathcal{F}_k \leftarrow \emptyset$; $gI \leftarrow \emptyset$;
foreach $attribute A_l \in db$ **do**
  /*build their respective gradual items:*/;
  $gI = gI + \{A_l\uparrow, A_l\downarrow\}$;
/* Candidate gradual itemset generation at level $k = 2$, of the form:*/;
$\mathcal{C} = \{A_1\uparrow A_2\uparrow, A_1\uparrow A_2\downarrow, ..., A_{m-1}\downarrow A_m\downarrow\}$;
foreach $gradual\ itemset\ candidate \in \mathcal{C}$ **do**
  $Compute\ their\ matrices\ of\ concordance\ degrees\ \tilde{cp}(i,j)$
  $Compute\ their\ support,\ as\ the\ sum\ of\ their\ matrices\ of\ concordance\ degrees\ divided\ by\ n(n-1)$;
  if $support(candidate) \geq minimum\_threshold$ **then**
    $\mathcal{F}_k \leftarrow \mathcal{F}_k \cup \{candidate\}$;
  Else $delete(candidate\ and\ matrix)$;
$k++$;
repeat
  /*Frequent gradual itemset mining,$k > 2$*/;
  $\mathcal{I} = GenItemset1From\{\mathcal{F}_{k-1}\}$;
  $\mathcal{J} = GenItemset2From\{\mathcal{F}_{k-1}\}$;
  $q = 1$;
  foreach $\{\mathcal{I}, \mathcal{J}\} \in \mathcal{F}_{k-1}$ **do**
    $Compute\ matrix\ of\ concordance\ degrees\ of\ candidate\ \mathcal{C}_{k,q}.M\ as:$
    $\mathcal{C}_{k,q}.M = \top - norm(\mathcal{I}.M, \mathcal{J}.M)$;
    /* $\mathcal{I}.M, \mathcal{J}.M$ are matrices of concordance degrees of itemsets $\mathcal{I}, \mathcal{J}$*/
    $support(\mathcal{C}_{k,q}) = sumMatrix(\mathcal{C}_{k,q}.M)/n(n-1)$;
    if $support(\mathcal{C}_{k,q}) \geq minimum\_threshold$ **then**
      $\mathcal{F}_k \leftarrow \mathcal{F}_k \cup \{\mathcal{C}_{k,q}\}$;
    Else $delete(candidate\ and\ matrix)$;
  $Delete(\mathcal{F}_{k-1}\ and\ Matrices)$;
  k++; $q++$;
until $\mathcal{F}_{FGP}$ *does not grow any more*;

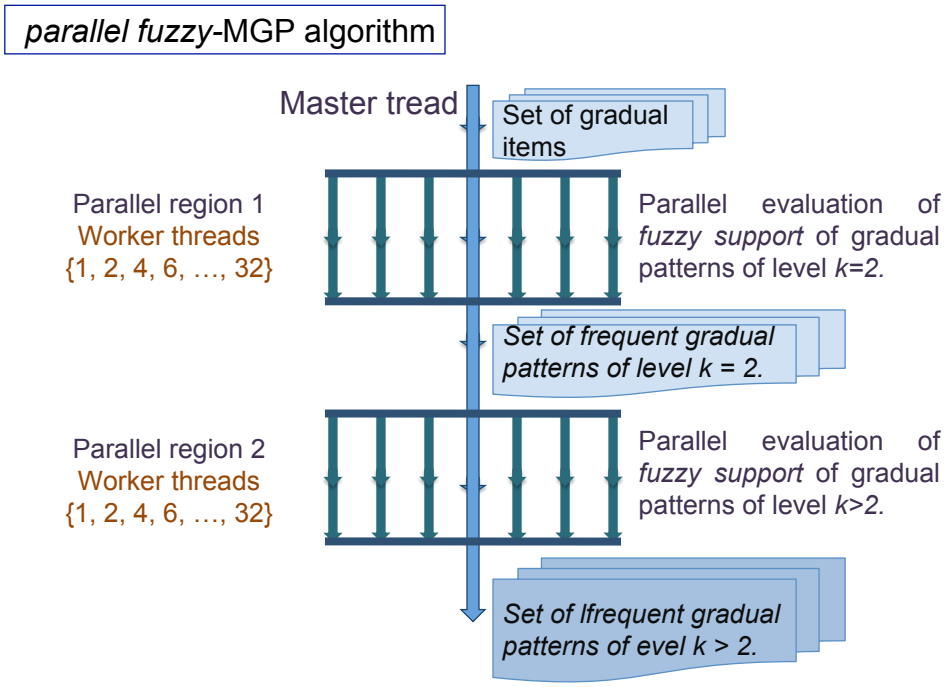
3.2. Coupling Multi-Precision and Parallel Programming

The evaluation of the correlation, support, and generation of gradual pattern candidates are tasks that require huge amounts of processing time, memory consumption, and load balance. In order to reduce memory consumption, each matrix of *fuzzy concordance degrees mc̃c(i,j)* is represented and

stored according to the *Binary Fuzzy Matrix Multi-precision Format*, where each $\tilde{cc}(i, j) \in [0, 1]$ is represented with a precision of 2, 3, or more up to 52 bits. In order to reduce processing time we propose to parallelize the program using OpenMP, a shared memory architecture API, which is ideally suited for multi-core architectures [9].

Figure 5 shows an overall view of the parallel version of two regions of our fuzzyMGP algorithm where, in the first region, the extraction process of gradual patterns of size $k = 2$ is parallelized. In the second region, we show the parallelization of the extraction cycle of gradual patterns of size $k > 2$.

Figure 5. Parallel extraction of gradual patterns (parfuzzyMGP).



In the experiments reported below, we aim at studying how multi-precision impacts performances, regarding the trade-off between high precision but high memory consumption and low memory consumption with low precision. The behavior of the algorithms is studied with respect to the number of bits allocated for storing the precision. The question raised is to study if there exists a threshold beyond which it is useless to consider allocating memory space. This threshold may depend on the database.

4. Experiments

4.1. Databases and Computing Resources

We lead experiments on two databases.

The first set of databases is a synthetic database generated in order to study scalability, and thus containing hundreds of attributes and lines that can be easily split in order to get several databases.

The second database comes from astrophysics called *Amadeus Exoplanete*, which consists of 97,718 instances and 60 attributes [10]. In this paper, we report experimental results of parallel gradual pattern mining from three subsets of data: of 1000, 2000, and 3000 instances with 15 attributes. The three datasets were obtained from *Amadeus Exoplanete* database.

In order to demonstrate the benefit of high performance computing on fuzzy data mining, our experiments are run on an IBM supercomputer, more precisely on two servers:

- an IBM dx360 M3 server embedding computing nodes configured with 2 $\times$ 2.66 GHx six core Intel (WESTMERE) processors, 24 Go DDR3 1,066 Mhz RAM and Infiniband (40 Gb/s) (reported as Intel); and
- an IBM x3850 X5 server running 8 processors embedding ten INTEL cores (WESTMERE), representing 80 cores at 2.26 GHz, 1 To DDR3 memory (1,066 Mhz) and Infiniband (40 Gb/s) reported as SMP (because of its shared memory).

4.2. Measuring Performances

In our experiments, we report the speedup of our algorithms regarding the database size and complexity [11]. Speedup is computed in order to prove the efficiency of our solution on high performance platforms and thus its scalability in order to tackle very large problems.

The *speedup* of a parallel program expresses the relative diminution of response time that can be obtained by using a parallel execution on p processors or cores compared to the best sequential implementation of that program. The *speedup* (*Speedup(p)*) of a parallel program with parallel execution time *T(p)* is defined as

$$Speedup(p) = \frac{T(1)}{T(p)} \tag{2}$$

where:

- p is the number of processors/cores or threads;
- T(1) is the execution time of the sequential program (with one thread or core);
- T(p) is the execution time of the parallel program with p processors, cores, or threads.

4.3. Main Results

We first notice that computing time is impacted by the choice of minimum threshold but it is not noticeably affected by a small difference of precision (see Figure 6). Furthermore it has no impact at all on measured speed-ups.

Figures 6–9 show that we can achieve very good accelerations on synthetic databases even for a relatively high level of parallelization (more than 50 processing units). In particular, Intel nodes show a good speedup on small precision (Figure 10), which shows the interest of managing multi-precision in order to adapt to the computing resources (memory) being available.

Regarding real database on astrophysics, our experiments show that on Intel nodes, 2000 lines can be managed at 4 bits precision (Figure 11), and up to 3000 lines at precision 2 bits (Figure 12) while it is impossible to manage 3000 lines at 4 bits precision due to memory consumption limits. On SMP nodes, our experiments show excellent speedup and scale up even over large databases, without memory explosion (Figure 13).

Figure 6. Execution time and Speedup related to the number of threads on Intel nodes for synthetic database of 150 attributes at precisions 6 and 9 bits.

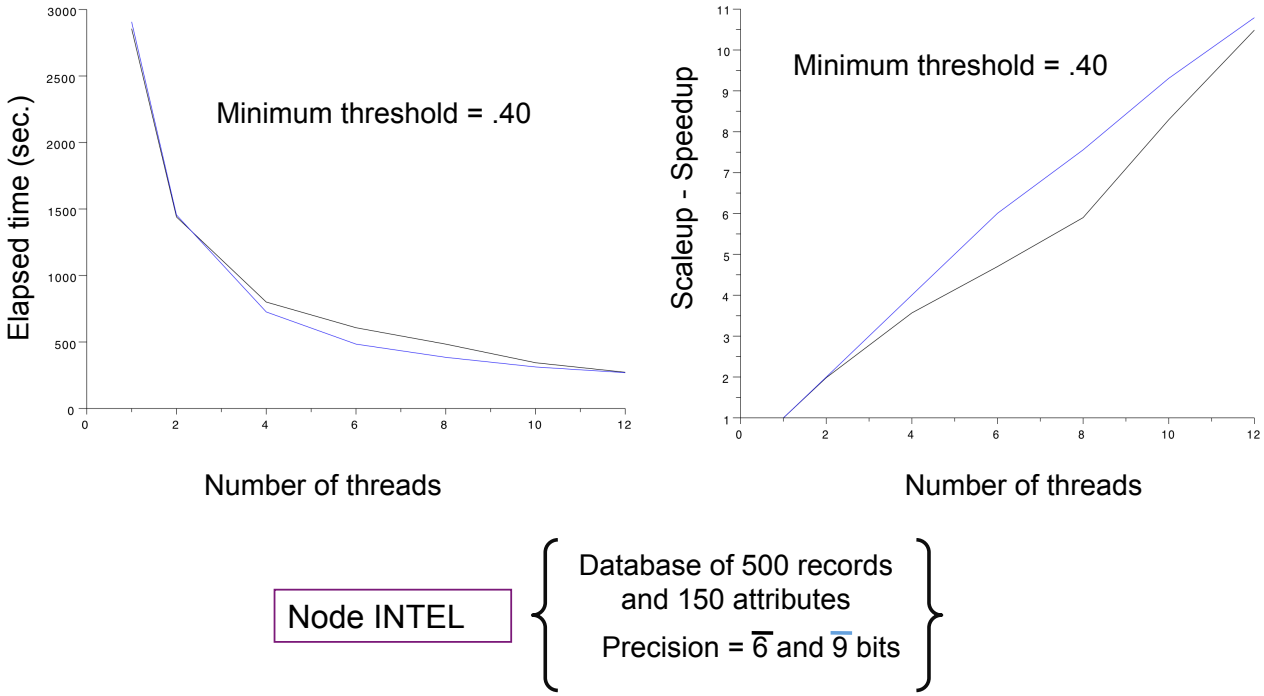


Figure 7. Execution time and Speedup related to the number of threads on Intel nodes for synthetic database of 200 attributes at precision 8 bits with minimum threshold values 0.411, 0.412 and 0.413.

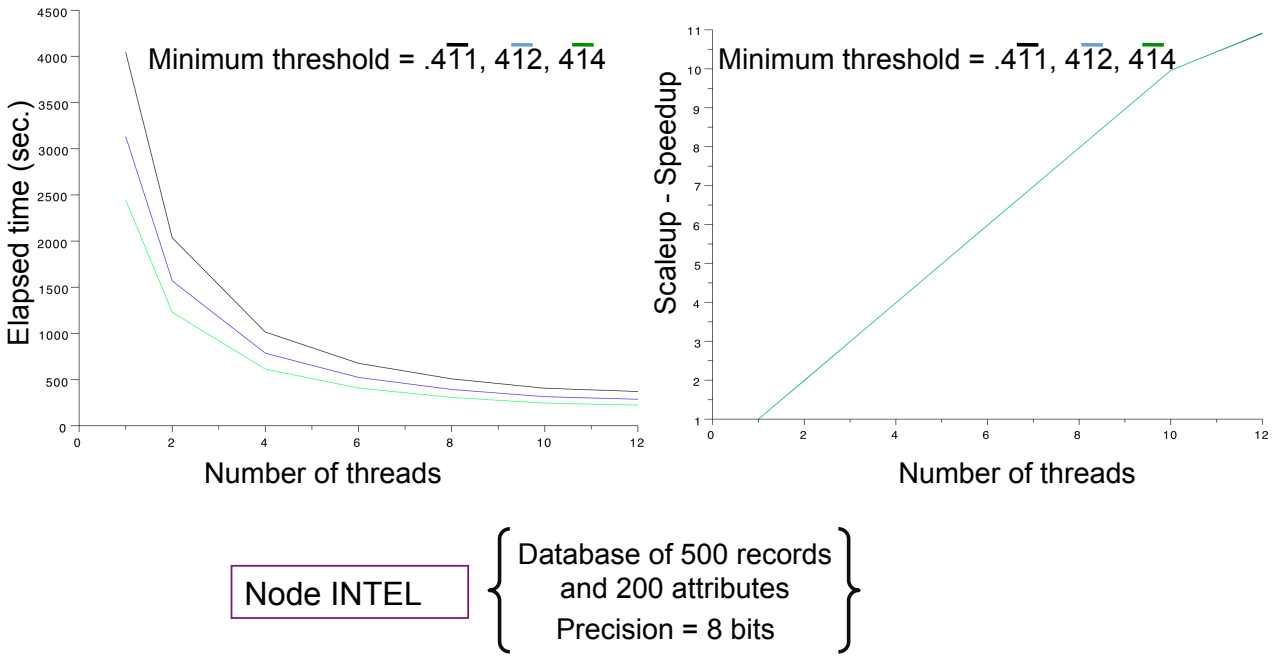


Figure 8. Execution time and Speedup related to the number of threads on SMP nodes for synthetic database of 200 attributes at precision 32 bits with minimum threshold values 0.411, 0.412, 0.414.

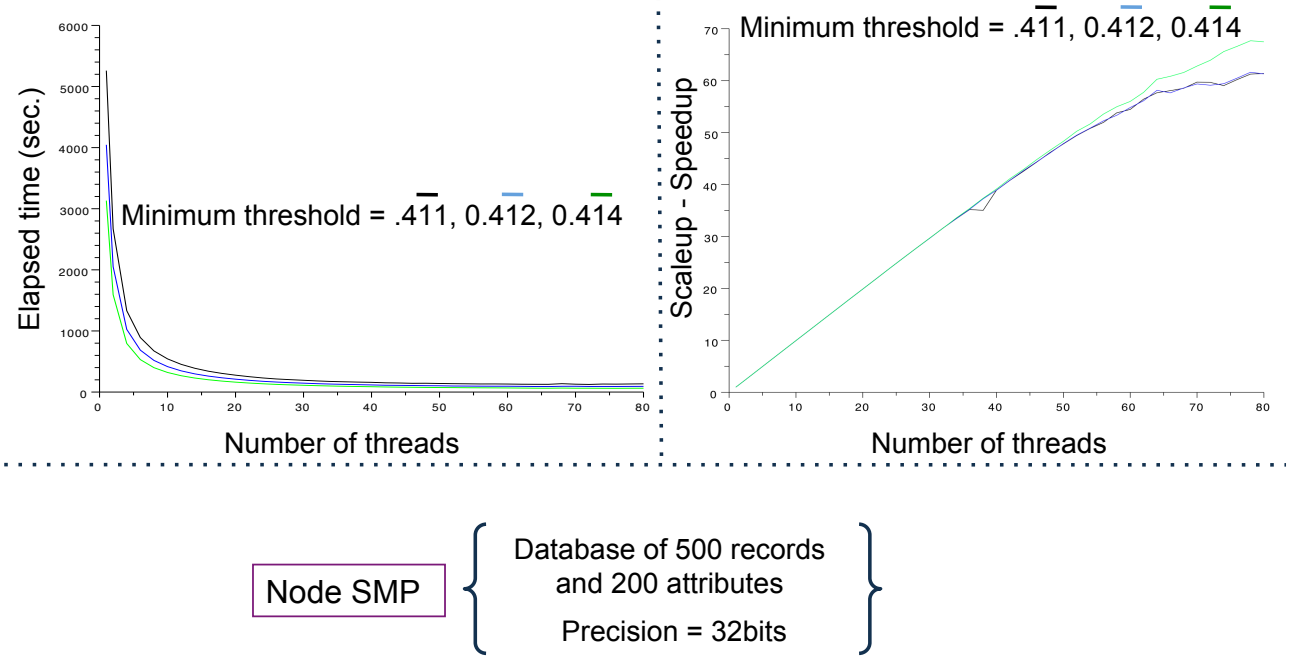


Figure 9. Execution time and Speedup related to the number of threads on SMP nodes for synthetic database of 300 attributes at precision 6 bits with minimum threshold values 0.4181 and 0.4183.

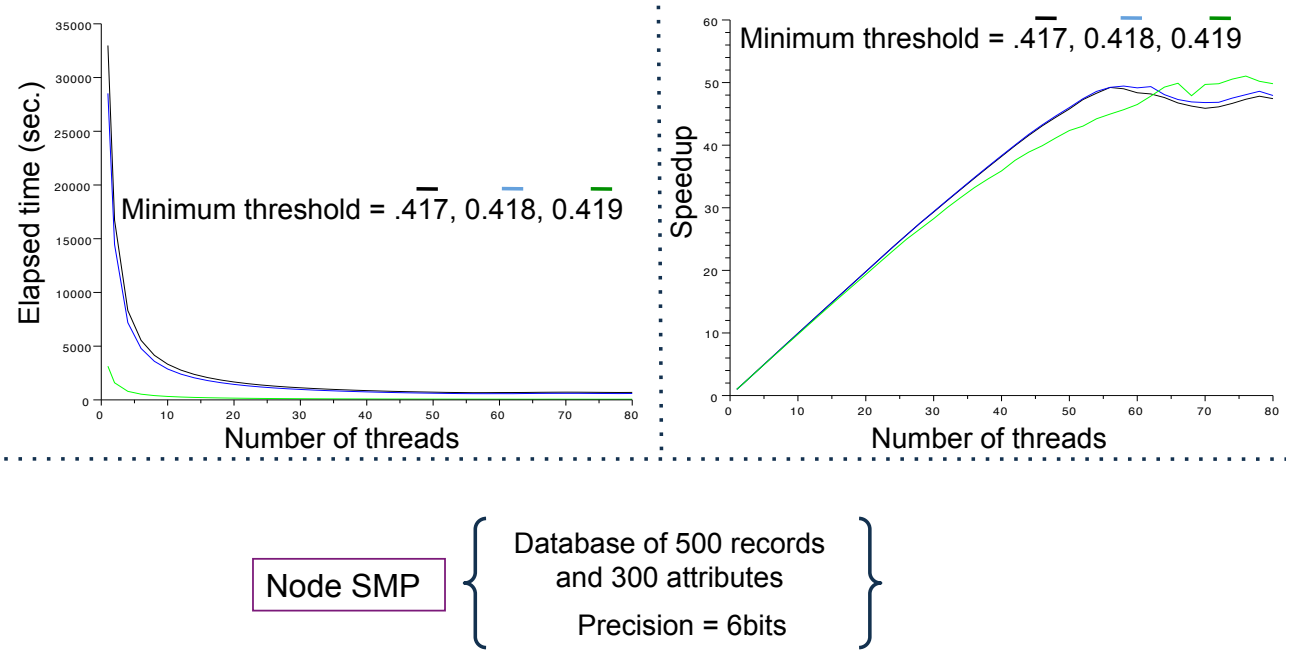


Figure 10. Execution time and Speedup related to the number of threads on Intel nodes for synthetic database of 300 attributes at precision 2 bits with minimum threshold values 0.17 and 0.18.

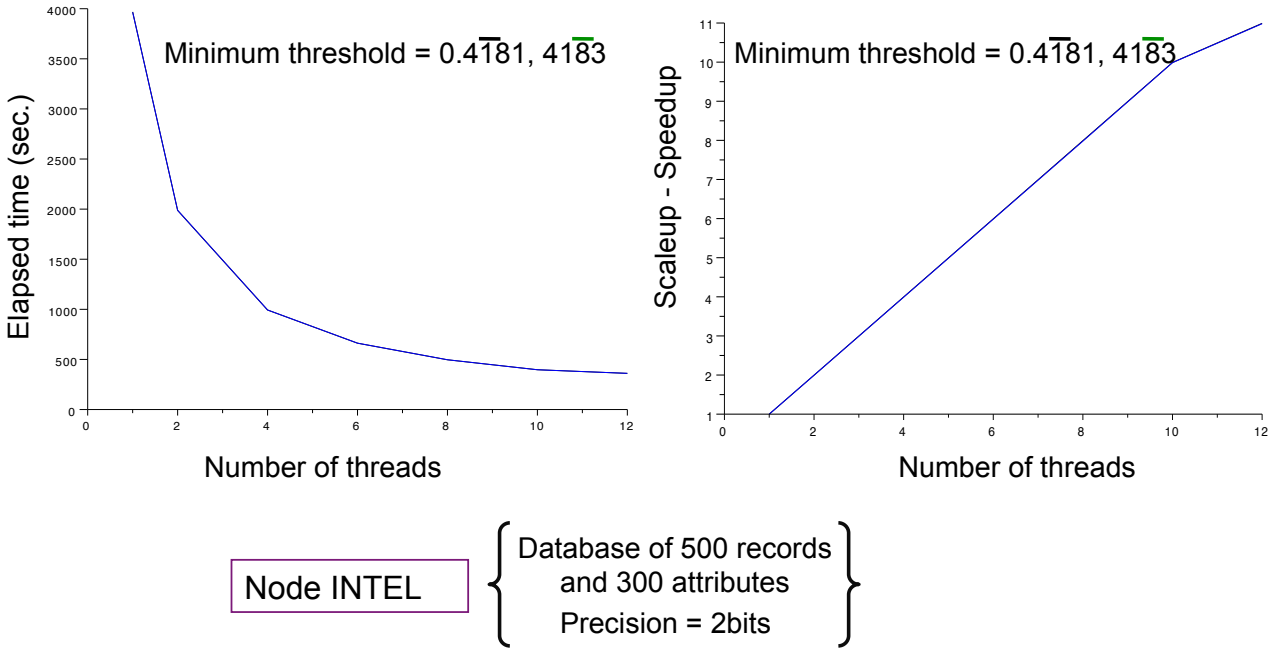


Figure 11. Execution time and Speedup related to the number of threads on Intel nodes for real database of 2000 lines at precision 4 bits with minimum threshold values 0.17 and 0.18.

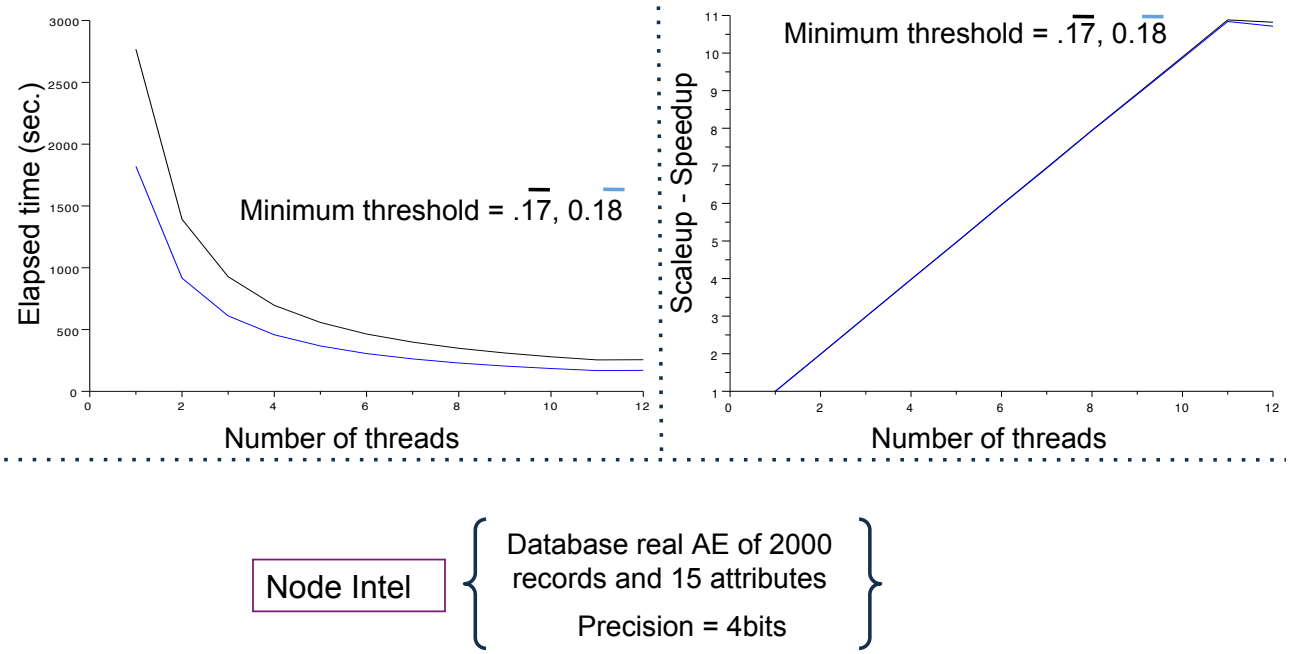


Figure 12. Execution time and Speedup related to the number of threads on Intel nodes for real database of 3000 lines at precision 2 bits with minimum threshold values 0.17 and 0.18.

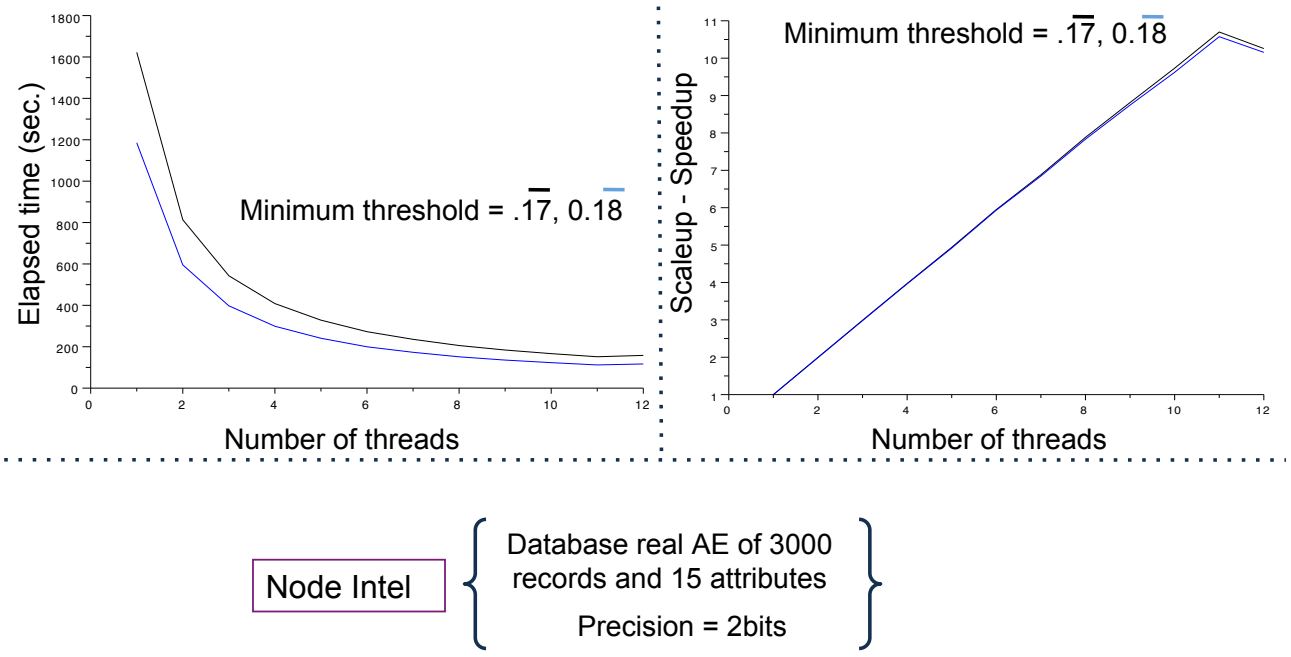
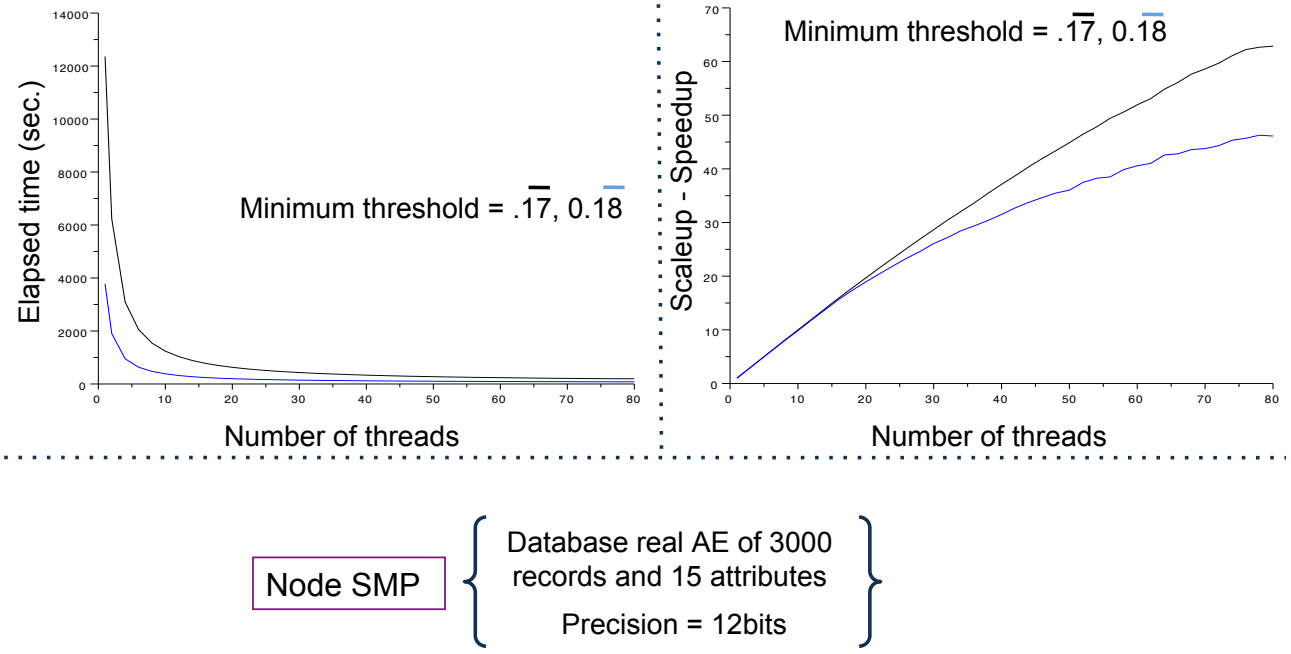


Figure 13. Execution time, Speedup and Scaleup related to the number of threads on SMP nodes for real database of 3000 lines at precision 12 bits with minimum threshold values 0.17 and 0.18.



5. Conclusions

In this paper, we address the extraction of gradual patterns when considering fuzzy ordering. This allows for dealing with imperfection in the datasets, when values can hardly be crisply ordered. For

instance, this situation often occurs when considering data collected from sensors. In this case, the measurement error leads to values that can be considered as being similar even if they are not equal. The extent to which they can be considered as similar is handled by considering *fuzzy orderings* and *fuzzy gamma rank correlation* which we propose to introduce in the gradual pattern mining algorithms. We show that the parallelization of such algorithms is necessary to remain scalable regarding both memory consumption and runtime. Memory consumption is indeed challenging in our framework as introducing fuzzy ranking prevents us to use a single bit for representing that such value is greater than such other one. We this introduce the notion of precision and we propose an efficient storage of the fuzzy concordance degrees that can be tuned (from 2 to 52 bits) in order to manage the trade-off between memory consumption and the loss or gain of computing power.

Acknowledgments

This work was realized with the support of HPC@LR, a Center of Competence in High-Performance Computing from the Languedoc-Roussillon region, funded by the Languedoc-Roussillon region, the Europe and the Université Montpellier 2 Sciences et Techniques. The HPC@LR Center is equiped with an IBM hybrid Supercomputer. The authors would also like to thank the AMADEUS CNRS MASTODONS project (Analysis of MAssive Data in Earth and Universe Sciences) for providing real data.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Bodenhofer, U. Fuzzy Orderings of Fuzzy Sets. In Proceedings of the 10th IFSA World Congress, Istanbul, Turkey, 30 June–2 July 2003; pp. 500–5007.
2. Koh, H.-W.; Hullermeier, E. Mining Gradual Dependencies Based on Fuzzy Rank Correlation. In *Combining Soft Computing and Statistical Methods in Data Analysis*; Volume 77, Advances in Intelligent and Soft Computing; Springer: Heidelberg, Germany, 2010; pp. 379–386.
3. Lin, N.P.; Chueh H. Fuzzy Correlation Rules Mining. In Proceedings of the 6th WSEAS International Conference on Applied Computer Science, Hangzhou, China, 15–17 April 2007; pp. 13–18.
4. Laurent, A.; Lesot, M.-J.; Rifqi, M. GRAANK: Exploiting Rank Correlations for Extracting Gradual Itemsets. In Proceedings of the Eighth International Conference on Flexible Query Answering Systems (FQAS'09), Volume LNAI 5822, Springer, Roskilde, Denmark, 26–28 October 2009; pp. 382–393.
5. Quintero, M.; Laurent, A.; Poncelet, P. Fuzzy Ordering for Fuzzy Gradual Patterns. In Proceedings of the FQAS 2011, Volume LNAI 7022, Springer, Ghent, Belgium, 26–28 October 2011; pp. 330–341.

6. Di Jorio, L.; Laurent, A.; Teisseire, M. Mining Frequent Gradual Itemsets from Large Databases. In Proceedings of the International Conference on Intelligent Data Analysis (IDA'09), Lyon, France, 31 August–2 September, 2009.
7. Quintero, M.; Laurent, A.; Poncelet, P.; Sicard, N. Fuzzy Orderings for Fuzzy Gradual Dependencies: Efficient Storage of Concordance Degrees. In Proceedings of the FUZZ-IEEE Conference, 10–15 June 2012, Brisbane, Australia.
8. El-Rewini, H.; Abd-El-Barr, M. *Advanced Computer Architecture Ans Parallel Processing*; Wiley: Hoboken, NJ, USA, 2005.
9. Rauber, T.; Rünger, G. *Parallel Programming: For Multicore and Cluster Systems*; Springer: Berlin/Heidelberg, Germany, 2010.
10. Debosscher, J.; Sarro, L.M. Automated supervised classification of variable stars in the CoRoT programme: Method and application to the first four exoplanet fields. *Astron. Astrophys.* **2009**, *506*, 519–534.
11. Hill, M.D. What is scalability? *ACM SIGARCH Comput. Archit. News* **1990**, *18*, 18–21.
12. Bodenhofer, U.; Klawonn, F. Roboust rank correlation coefficients on the basis of fuzzy orderings: Initial steps. *Mathw. Soft Comput.* **2008**, *15*, 5–20.
13. Calders, T.; Goethais, B.; Jarszewicz, S. Mining Rank-Correlated Sets of Numerical Attributes. In Proceedings of the KDD'06, 20–23 August 2006; ACM: Philadelphia, PA, USA, 2006.
14. Flynn, M. Some computer organizations and their effectiveness. *IEEE Trans. Comput.* **1972**, *C-21*, 948–960.
15. Hüllermeier, E. Association Rules for Expressing Gradual Dependencies. In Proceedings of the PKDD Conference, 19–23 August 2002; Volume LNCS 2431, Helsinki, Finland; pp. 200–211.
16. Zadeh, L.A. Similarity relations and fuzzy orderings. *Inf. Sci.* **1971**, *3*, 177–200.