M2LFGP: Mining Gradual Patterns over Fuzzy Multiple Levels

Yogi Satrya Aryadinata, Arnaud Castelltort, Anne Laurent, Michel Sala

M2LFGP: Mining Gradual Patterns over Fuzzy Multiple Levels

Yogi S. Aryadinata, Arnaud Castelltort, Anne Laurent, and Michel Sala

University Montpellier 2
LIRMM - CNRS UMR 5506,
161, Rue Ada, 34392 Montpellier Cedex 5, France
{yogi.aryadinata,arnaud.castelltort,
laurent,michel.sala}@lirmm.fr
http://www.lirmm.fr

Abstract. Data are often described at several levels of granularity. For instance, data concerning fruits that are purchased can be categorized regarding some criteria (such as size, weight, color, etc.). When dealing with data from the real world, such categories can hardly be defined in a crisp manner. For instance, some fruits may belong both to the *small* and *medium*-sized fruits. Data mining methods have been proposed to deal with such data, in order to take benefit from the several levels when extracting relevant patterns. The challenge is to discover patterns that are not too general (as they would not contain relevant novel information) while remaining typical (as detailed data do not embed general and representative information). In this paper, we focus on the extraction of gradual patterns in the context of hierarchical data. Gradual patterns describe covariation of attributes such as *the bigger, the more expensive*. As our proposal increases the number of combinations to be considered since all levels must be explored, we propose to implement the parallel computation in order to decrease the execution time.

1 Introduction

Databases are often considered in order to extract relevant information that describe the patterns occurring. For instance, gradual patterns such as *the bigger, the more expensive* can be extracted from data describing purchases of fruits.

Example 1. As presented in [1], we consider the database containing sales from a shop selling fruits as shown in Table 1. For the sake of simplicity, we consider here natural numbers (e.g., number of kg) but our approach also works on other domains, provided the fact that they are provided with a partial order Each tuple from the database corresponds to a cashier ticket.

Gradual patterns are extracted from databases where data can be ordered (e.g., numeric databases).

Several algorithms have been proposed for discovering such gradual patterns, based on the ones that have been proposed in the literature. These algorithms

H.L. Larsen et al. (Eds.): FQAS 2013, LNAI 8132, pp. 437–446, 2013.

Table 1. Fruit Sales Database

Id	Pineapple	RedApples	Cherries	Durian
T1	0	3	0	0
T2	2	1	1	2
T3	4	4	2	3
T4	2	1	1	1
T5	7	0	3	0

have been studied either to optimise performances (in terms of memory and/or runtimes) [2–4], or to discuss various manners of computing to which extent a pattern is present in a database or which patterns and rules can be discovered [5–8].

Fuzzy extensions have been defined in order to deal with real life applications where data and knowledge are often not crisp. [9] studies the possibility that the graduality is not over all the attribute but may be hidden somewhere in the domain of values. For instance, when mining gene expression, it may be the case that there is no pattern such as "The more the expression of gene G_i, the less the expression of gene G_j" over the whole interval but that it is rather the case that the pattern correlates values within the interval of values. Such a pattern may be "The more the expression of gene G_i is *almost* 0.2, the less the expression of gene G_j is *almost* 0.8". The approach proposes a definition of such fuzzy patterns and algorithms based on genetic programming in order to discover the most relevant parts of the universe (e.g. *almost* 0.2 and *almost* 0.8 *in the above example).*

[10] proposes to consider fuzzy orders instead of crisp orders so as to tackle the problem of data where differences between values may not always convey a crisp decision. For instance, it may be the case that an expression of gene of 0.1887 may not be that lower than an expression of gene of 0.1888.

More recently, we have studied how hierarchies can be managed in order to discover patterns at several levels of granularity, based on the work from the literature addressing multiple level data mining [11, 12].

Hierarchies can be either horizontal or vertical.

- horizontal hierarchies allow to merge several columns, for instance to put together small fruits,
- vertical hierarchies allow to merge several lines, for instance to merge purchases made within the same day.

Example 2. In Table 2, we consider the database from Example 1, where T1, T2 and T3 are from Monday and T4 and T5 are from Tuesday.

Table 2. Vertical Aggregation

Id	Pineapple	RedApples	Cherries	Durian
Monday (T1-3)	6	8	3	5
Tuesday (T4-5)	9	1	4	1

Example 3. In Table 3, we consider that the first two columns may be merged.

Table 3. Horizontal Aggregation

Id	WithoutKernel (Pineapple + RedApples)	WithKernel (Cherries)
T1	3	0
T2	3	1
T3	8	2
T4	3	1
T5	7	3

Several hierarchies can be defined.

Table 4. RedFruit Aggregation

	NotRed (Pineapple)	Red (RedApples + Cherries)
T1	0	3
T2	2	2
T3	4	6
T4	2	2
T5	7	3

However, real world data are often described by fuzzy hierarchies (e.g., a people can hardly be always crisply categorized into “young” or “old”, a city can hardly be always crisply categorised into “south” and “north”).

In this paper, we thus study how such fuzzy hierarchies can help for discovering relevant gradual patterns at multiple levels of granularity. As it introduces computation complexity, we consider the use of supercomputers in order to remain efficient and scalable over large and complex databases.

2 Gradual Patterns: Preliminary Definitions

Gradual patterns have been studied in the literature. Several notations have been proposed, we have chosen to consider the ones given below. Roughly speaking, gradual patterns are extracted by discovering the largest subsets of data that can be ordered when considering the corresponding attributes. For instance, from Example 1, we may consider the pattern "The higher the number of pineapples, the higher the number of durians" since the first four tuples can be ordered as $< T_1, T_4, T_2, T_3 >$ with $T_1.Pineapple \leq T_1.Durian$ and $T_4.Pineapple \leq T_4.Durian$ and $T_2.Pineapple \leq T_2.Durian$ and $T_3.Pineapple \leq T_3.Durian$.

Definition 1. *Gradual-Attribute. A gradual attribute I is defined over a domain $dom(I_j)$ on which an order $\leq_j$ (or simply $\leq$) is defined.*

Definition 2. *Gradual-DB. A gradual database is a set of tuples $\mathcal{T}$ defined over the schema $\mathcal{S} = \{Id, I_1, \ldots, I_n\}$ of n gradual attributes where Id is an identifier (primary key).*

Example 1 shows an example of a database which schema is S=$\{Pineapple$, $RedApples, Cherries, Durian\}$ containing 5 tuples defined over three attributes which domains are $\mathbb{N}$.

Definition 3. *Gradual item. A gradual item is a pair (i, v) where i is an item and v is variation $v \in \{\uparrow, \downarrow\}$. $\uparrow$ stands for an increasing variation while $\downarrow$ stands for a decreasing variation.*

For example, $(Pineapple, \uparrow)$ is a gradual item.

Definition 4. *Gradual Pattern (also known as Gradual Itemset). A gradual pattern is a set of gradual items, denoted by $GP = \{(i_1, v_1), \ldots, (i_n, v_n)\}$. The set of all gradual patterns that can be defined is denoted by GP.*

For example, $\{(Pineapple, \uparrow), (RedApples, \uparrow)\}$ is a gradual itemset.

Definition 5. *Tuple Ordering Over a Set of Attributes A. The tuples from a gradual database are ordered by defining an order $\prec$ with respect to a gradual pattern $GP\{(i_1, v_1), \ldots, (i_n, v_n)\}$. Two tuples t and t' can be ordered with respect to GP, denoted by $t \prec_{GP} t'$ if all the values of the corresponding items can be ordered with respect to the variations: for every $i_k (k \in [1, n])$, $t.i_k \leq t'.i_k$ if $v_l = \uparrow$ and $t'.i_k \leq t.i_k$ if $v_l = \downarrow$.*

When mining for gradual patterns, the goal is to extract frequent patterns. We thus have to determine what *frequent* means.

Definition 6. *Gradual Support. The support of a gradual pattern over a gradual database GDB is a function supp from $\mathcal{GP}$ to $[0, 1]$ that holds the following property (anti-monotonicity): for all GP_1, $GP_2 \in \mathcal{GP}$, $GP_1 \subseteq GP2 \Rightarrow supp(GP_1) \geq supp(GP_2)$. The support is $support(GP) = \frac{max_{L \in l}(|L|)}{(|R|)}$, where is the longest list of tuples that respects the gradual itemset of GP and $L = t_1, t_2, ..., t_m$ is a list of tuples from a set of tuples R.*

Definition 7. *Frequent Gradual Pattern. Given a "minimum support" threshold σ, a gradual pattern GP is said to be frequent if $supp(GP) \geq \sigma$.*

When dealing with hierarchies, we consider the case where columns or lines can be merged regarding their belonging to a common category in a taxonomy as shown in Table 4.

A multiple level attribute MLA is an attribute equipped with a hierarchy. This hierarchy is defined as a set of levels where every level is represented as a partition, all the partitions being embedded.

Definition 8. *ML-Attribute. An ML-Attribute MLA is defined by:*

- *a label,*
- *a domain $dom(MLA)$,*
- *a set of embedded levels of granularity $\mathcal{L} = L_0(MLA), \ldots, L_g(MLA)$ such as every domain $dom(L_i)$ is ordered with a relation $\leq_{L_i}$.*

Frequent gradual patterns are extracted from such databases by building, as described in [1], all possible hierarchies from the raw data. MLGP Operators are considered to aggregate values when merging columns and lines. For instance, the MLGP operator can be denoted by $\oplus$ and values are summed up as in the following example. In Table 3, results are built by summing up values for Pineapples and RedApples in the one hand and Cherries in the other hand.

3 M2LFGP: Dealing with Fuzzy Hierarchies

When dealing with fuzzy hierarchies, we consider that an item can be embedded within several upper categories at some extent, this extent being expressed by a degree ranging from 0 to 1 [13].

For instance, fruits belong to the category of small or big fruits at a certain extent, as described on Fig. 1. It should be noted that we do not require that all the degrees from a given category sum up to 1.

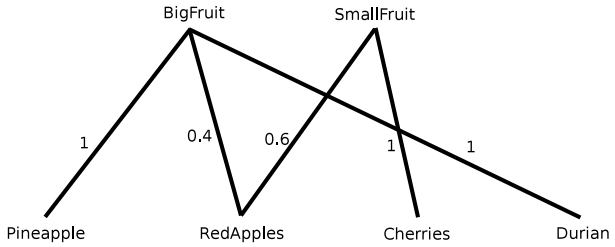

Fig. 1. Horizontal Hierarchy: Fruit Size

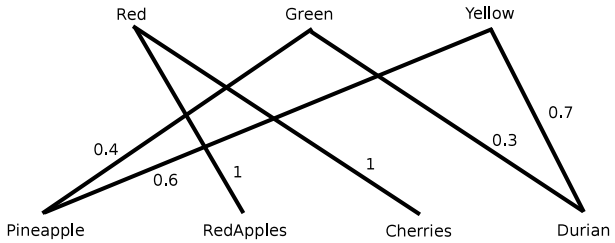

Fig. 2. Horizontal hierarchy: Fruit Color

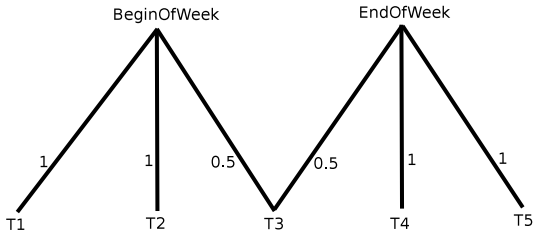

Fig. 3. Vertical Hierarchy: TimeStamps

When dealing with such cases, values must be merged differently from the crisp case. In order to take fuzzy hierarchies into account, we consider an MFLGP Operator which fuzzifies MLGP Operators. For this purpose, values are merged by integrating the degree from the hierarchy.

For this purpose, n-ary operations built from T-norms are considered.

For instance, we consider the product T-norm and the sum as the MLGP operator $\oplus$.

In the following table, we consider an example that merges lines for Begin of week and End of Week.

Table 5. Vertical Fuzzy Aggregation

Id	Pineapple	RedApples	Cherries	Durian
BeginOfWeek	4	6	2	3.5
EndOfWeek	11	3	3	2.5

In this example, the value for Durian for "BeginOfWeek" is 3.5, computed as: $0 * 1 + 2 * 1 + 3 * 0.5 = 3.5$.

When merging lines with an horizontal hierarchy, we consider the computation of the merged columns by considering the degree D of an item value v to be

mapped with hierarchy value h as $D = v * degree_h$. These degrees are then merged using the $\oplus$ operator.

For instance, the following example shows how columns can be merged for small and big fruits with $\oplus = +$:

Table 6. Horizontal Fuzzy Aggregation

Id	BigFruit	SmallFruit
T1	1.2	1.8
T2	4.4	1.6
T3	8.6	5.4
T4	2.4	1.6
T5	7	3

In this example, the value for SmallFruit for tuple 1 is 1.8, computed as: $3 * 0.6 + 0 * 1 = 1.8$.

4 M2LFGP: Algorithms

In this part, we show how to extend our previous work to handle fuzziness. The algorithms being considered are given below. They allow to build databases from the source database provided with fuzzy hierarchies. Roughly speaking, the algorithms must combine fuzzy horizontal and vertical hierarchies.

The databases being generated from such transformations are then mined by classical algorithms for discovering relevant gradual patterns.

When generating the databases, the introduction of fuzzy hierarchies requires both to:

- represent fuzzy degrees;
- take the degrees into account when transforming databases for considering hierarchies.

Fuzzy hierarchies are represented using an XML description, as shown by Fig. 4.

Every hierarchy, should it be horizontal or vertical, is taken into account for transforming the database, in associated operations: Horizontal transformation, Vertical transformation and Horizontal-Vertical transformation which merge columns or lines regarding the definition of the hierarchy.

5 M2LFGP: Experimental Results

In our experimentation, the algorithm is implemented in Java for the dataset preparation / transformation and C++ for the parallel gradual pattern mining algorithm [3]. We study the efficiency of our method to get more valuable results.

```
<graph>
  <node id="n0">
    <name>BigFruit</name>
    <level>1</level>
  </node>
  <node id="n1">
    <name>SmallFruit</name>
    <level>1</level>
  </node>
  <node id="n2" >
    <name>PineApple</name>
    <level>0</level>
  </node>
  .
  .
  <edge id="e0" source="n0" target="n2">
    <weight>0.4</weight>
  </edge>
  <edge id="e1" source="n1" target="n2">
    <weight>0.6</weight>
  </edge>
  .
  .
</graph>
```

Fig. 4. XML Representation of a Fuzzy Hierarchy

In order to decrease runtimes, we consider parallel programming and we run our code on a server proposing up to 32 processing cores. This server provides a 8 AMD Opteron 852 (every processor being provided with 4 cores), 64 GB of RAM running Linux Centos 5.

We consider a synthetic database integrating hierarchies. The hierarchies are automatically generated from the databases. The dataset, C150A30, contains 150 tuples and 30 attributes, with 3 generated hierarchies (2 horizontal hierarchies and 1 vertical hierarchies). The horizontal hierarchies that have been considered contain 2 top level nodes and 3 top level nodes. The vertical hierarchy contains 2 top level nodes.

Figures 5 and 6 show the evolution of runtimes and speed up. Speed up displays how runtime can decrease when several processors and cores are considered. The more linear the speed up, the better, showing that runtimes decreases when the number of threads increases. For example, the comparison between sequential and parallel executions shows that sequential execution takes about 350 seconds while the runtime with 2 threads is reduced down to 168 seconds, as well the runtime with 3 threads that goes down to 80 seconds. As we can see, the execution time is greatly reduced until 4 threads. This happens because the processing of dataset has nearly reached the most efficient number of threads.

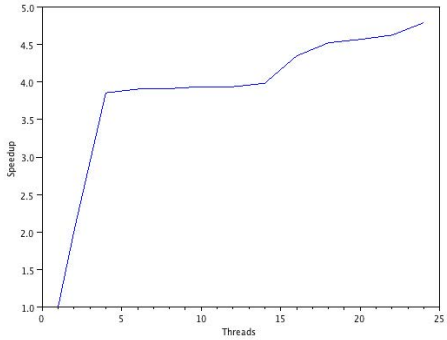

Fig. 5. Speed Up

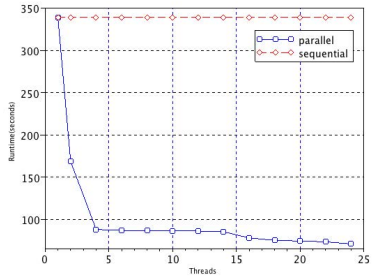

Fig. 6. Runtime over the number of threads

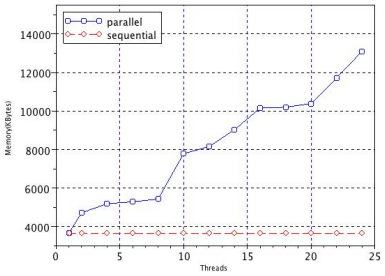

Fig. 7. Memory usage over the number of threads

The memory consumption is very low regarding the number of threads and the size of our dataset, as shown by Figure 7. In order to produce a better result, we need to improve our experiments. In particular, we plan to use the real-world datasets from environmental domain.

6 Conclusion and Future Work

In this paper, we propose the original method M2LFGP for mining relevant gradual patterns over fuzzy multiple levels (hierarchies). This work is very important as many real world databases contain fuzzy hierarchies in order to describe the data. We provide the necessary definitions together with algorithms that have been tested through experiments.

Future works include the study of the discovery of fuzzy gradual patterns (i.e., fuzzy items) over fuzzy hierarchies. Our work can also be improved by studying how the properties of the fuzzy hierarchy (e.g., degrees summing up to 1) can be exploited in order to design more efficient algorithms. Moreover, we aim at

further studying scalability by using the parallel programming paradigms, especially for taking benefit from high performance architectures that are specialised in data distribution. Finally, we aim at applying our algorithms on several real databases in order to prove its relevance.

References

1. Laurent, A., Aryadinata, Y., Sala, M.: M2LGP: Mining multiple level gradual patterns. In: Proc. of ICKDDM 2013: International Conference on Knowledge Discovery and Data Mining (2013)
2. Di-Jorio, L., Laurent, A., Teisseire, M.: Mining frequent gradual itemsets from large databases. In: Adams, N.M., Robardet, C., Siebes, A., Boulicaut, J.-F. (eds.) IDA 2009. LNCS, vol. 5772, pp. 297–308. Springer, Heidelberg (2009)
3. Do, T.D.T., Laurent, A., Termier, A.: PGLCM: Efficient parallel mining of closed frequent gradual itemsets. In: Proc. of ICDM 2010, The 10th IEEE International Conference on Data Mining, pp. 138–147 (2010)
4. Laurent, A., Negrevergne, B., Sicard, N., Termier, A.: PGP-mc: Towards a multicore parallel approach for mining gradual patterns. In: Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C. (eds.) DASFAA 2010. LNCS, vol. 5981, pp. 78–84. Springer, Heidelberg (2010)
5. Hüllermeier, E.: Association rules for expressing gradual dependencies. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) PKDD 2002. LNCS (LNAI), vol. 2431, pp. 200–211. Springer, Heidelberg (2002)
6. Berzal, F., Cubero, J.C., Sanchez, D., Vila, M.A., Serrano, J.M.: An alternative approach to discover gradual dependencies. Int. Journal of Uncertainty, Fuzziness and Knowledge-Based Systems (IJUFKS) 15(5), 559–570 (2007)
7. Laurent, A., Lesot, M.-J., Rifqi, M.: GRAANK: Exploiting rank correlations for extracting gradual itemsets. In: Andreasen, T., Yager, R.R., Bulskov, H., Christiansen, H., Larsen, H.L. (eds.) FQAS 2009. LNCS, vol. 5822, pp. 382–393. Springer, Heidelberg (2009)
8. Bouchon-Meunier, B., Laurent, A., Lesot, M.J., Rifqi, M.: Strengthening fuzzy gradual rules through "all the more" clauses. In: FUZZ-IEEE (2010)
9. Ayouni, S., Yahia, S.B., Laurent, A., Poncelet, P.: Fuzzy gradual patterns: What fuzzy modality for what result? In: Proc. of the Second International Conference of Soft Computing and Pattern Recognition (SoCPaR), pp. 224–230. IEEE (2010)
10. Quintero, M., Laurent, A., Poncelet, P.: Fuzzy orderings for fuzzy gradual patterns. In: Christiansen, H., De Tré, G., Yazici, A., Zadrozny, S., Andreasen, T., Larsen, H.L. (eds.) FQAS 2011. LNCS, vol. 7022, pp. 330–341. Springer, Heidelberg (2011)
11. Han, J., Fu, Y.: Mining multiple-level association rules in large databases. IEEE Trans. Knowl. Data Engin. 11(5), 798–804 (1999)
12. Plantevit, M., Laurent, A., Laurent, D., Teisseire, M., Choong, Y.W.: Mining multidimensional and multilevel sequential patterns. ACM Transactions on Knowledge Discovery from Data (TKDD) 4(1) (2010)
13. Buckley, J.J., Feuring, T., Hayashi, Y.: Fuzzy hierarchical analysis revisited. European Journal of Operational Research 129(1), 48–64 (2001)