Interestingness is not a Dichotomy: Introducing Softness in Constrained Pattern Mining

Stefano Bistarelli[1,2] and Francesco Bonchi[3]

[1] Dipartimento di Scienze, Università degli Studi "G. D'Annunzio", Pescara, Italy
[2] Istituto di Informatica e Telematica, CNR, Pisa, Italy
[3] Pisa KDD Laboratory, ISTI - C.N.R., Pisa, Italy
e-mail: bista@sci.unich.it; francesco.bonchi@isti.cnr.it

Abstract. The paradigm of pattern discovery based on constraints was introduced with the aim of providing to the user a tool to drive the discovery process towards potentially *interesting* patterns, with the positive side effect of achieving a more efficient computation. So far the research on this paradigm has mainly focussed on the latter aspect: the development of efficient algorithms for the evaluation of constraint-based mining queries. Due to the lack of research on methodological issues, the constraint-based pattern mining framework still suffers from many problems which limit its practical relevance. As a solution, in this paper we introduce the new paradigm of pattern discovery based on *Soft Constraints*. Albeit simple, the proposed paradigm overcomes all the major methodological drawbacks of the classical constraint-based paradigm, representing an important step further towards practical pattern discovery.

1 Background and Motivations

During the last decade a lot of researchers have focussed their (mainly algorithmic) investigations on the computational problem of *Frequent Pattern Discovery*, i.e. mining patterns which satisfy a user-defined constraint of minimum frequency [1].

The simplest form of a frequent pattern is the frequent itemset.

Definition 1 (Frequent Itemset Mining). *Let $\mathcal{I} = \{x_1, ..., x_n\}$ be a set of distinct items, where an item is an object with some predefined attributes (e.g., price, type, etc.). An itemset X is a non-empty subset of $\mathcal{I}$. A transaction database $\mathcal{D}$ is a bag of itemsets $t \in 2^{\mathcal{I}}$, usually called transactions. The support of an itemset X in database $\mathcal{D}$, denoted $supp_{\mathcal{D}}(X)$, is the number of transactions which are superset of X. Given a user-defined minimum support σ, an itemset X is called frequent in $\mathcal{D}$ if $supp_{\mathcal{D}}(X) \geq \sigma$. This defines the minimum frequency constraint: $\mathcal{C}_{freq[\mathcal{D},\sigma]}(X) \Leftrightarrow supp_{\mathcal{D}}(X) \geq \sigma$.*

Recently the research community has turned its attention to more complex kinds of frequent patterns extracted from more structured data: *sequences*, *trees*, and *graphs*. All these different kinds of pattern have different peculiarities and application fields, but they all share the same computational aspects: a usually very large input, an exponential search space, and a too large solution set. This situation – too many data yielding too many patterns – is harmful for two reasons. First, performance degrades: mining generally becomes inefficient or, often, simply unfeasible. Second, the identification of the

fragments of interesting knowledge, blurred within a huge quantity of mostly useless patterns, is difficult. The paradigm of *constraint-based pattern mining* was introduced as a solution to both these problems. In such paradigm, it is the user which specifies to the system what is *interesting* for the current application: constraints are a tool to drive the mining process towards potentially interesting patterns, moreover they can be pushed deep inside the mining algorithm in order to fight the exponential search space curse, and to achieve better performance [15, 20, 25].

When instantiated to the pattern class of itemsets, the constraint-based pattern mining problem is defined as follows.

Definition 2 (Constrained Frequent Itemset Mining). *A constraint on itemsets is a function $\mathcal{C} : 2^{\mathcal{I}} \rightarrow \{true, false\}$. We say that an itemset I satisfies a constraint if and only if $\mathcal{C}(I) = true$. We define the $theory$ of a constraint as the set of itemsets which satisfy the constraint: $Th(\mathcal{C}) = \{X \in 2^{\mathcal{I}} \mid \mathcal{C}(X)\}$. Thus with this notation, the frequent itemsets mining problem requires to compute the set of all frequent itemsets $Th(\mathcal{C}_{freq[\mathcal{D},\sigma]})$. In general, given a conjunction of constraints $\mathcal{C}$ the constrained frequent itemsets mining problem requires to compute $Th(\mathcal{C}_{freq}) \cap Th(\mathcal{C})$.*

Example 1. The following is an example mining query:

$$\mathcal{Q}: \ supp_{\mathcal{D}}(X) \geq 1500 \ \wedge \ avg(X.weight) \leq 5 \ \wedge \ sum(X.price) \geq 20$$

It requires to mine, from database $\mathcal{D}$, all patterns which are frequent (have a support larger than 1500), have average weight less than 5 and a sum of prices greater than 20.

So far constraint-based frequent pattern mining has been seen as a query optimization problem, i.e., developing efficient, sound and complete evaluation strategies for constraint-based mining queries. Or in other terms, designing efficient algorithms to mine all and only the patterns in $Th(\mathcal{C}_{freq}) \cap Th(\mathcal{C})$. To this aim, properties of constraints have been studied comprehensively, and on the basis of such properties (e.g., anti-monotonicity, succinctness [20, 18], monotonicity [11, 17, 6], convertibility [22], loose anti-monotonicity [9]), efficient computational strategies have been defined. Despite such effort, the constraint-based pattern mining framework still suffers from many problems which limit its practical relevance.

First of all, consider the example mining query $\mathcal{Q}$ given above: *where do the three thresholds (i.e., 1500, 5 and 20) come from?* In some cases they can be precisely imposed by the application, but this is rarely the case. In most of the cases, they come from an exploratory mining process, where they are iteratively adjusted until a solution set of reasonable size is produced. This practical way of proceeding is in contrast with the basic philosophy of the constraint-based paradigm: constraints should represent what is a priori interesting, given the application background knowledge, rather than be adjusted accordingly to a preconceived output size. Another major drawback of the constraint-based pattern mining paradigm is its rigidity. Consider, for instance, the following three patterns (we use the notation $\langle v_1, v_2, v_3 \rangle$ to denote the three values corresponding to the three constraints in the conjunction in the example query $\mathcal{Q}$): $p_1 : \langle 1700, 0.8, 19 \rangle$, $p_2 : \langle 1550, 4.8, 54 \rangle$, and $p_3 : \langle 1550, 2.2, 26 \rangle$. The first pattern, p_1, largely satisfies two out of the three given constraints, while slightly violates the third one. According to

the classical constraint-based pattern mining paradigm p_1 would be discarded as non interesting. Is such a pattern really *less interesting* than p_2 and p_3 which satisfy all the three constraints, but which are much less frequent than p_1? Moreover, is it reasonable, in real-world applications, that all constraints are equally important?

All these problems flow out from the same source: the fact that in the classical constraint-based mining framework, a constraint is a function which returns a boolean value $\mathcal{C} : 2^{\mathcal{I}} \rightarrow \{true, false\}$. Indeed, *interestingness is not a dichotomy*.

This consideration suggests us a simple solution to overcome all the main drawbacks of constraint-based paradigm.

Paper Contributions and Organization

In this paper, as a mean to handle interestingness [26, 16, 24], we introduce the *soft constraint based pattern mining* paradigm, where constraints are no longer rigid boolean functions, but are "soft" functions, i.e., functions with value in a set A, which represents the set of interest levels or costs assigned to each pattern.

- The proposed paradigm is not rigid: a potentially interesting pattern is not discarded for just a slight violation of a constraint.
- Our paradigm creates an order of patterns w.r.t. interestingness (level of constraints satisfaction): this allows to say that a pattern is *more interesting* than another, instead of strictly dividing patterns in interesting and not interesting.
- From the previous point it follows that our paradigm allows to express *top-k* queries based on constraints: the data analyst can ask for the top-10 patterns w.r.t. a given description (a conjunction of soft constraints).
- Alternatively, we can ask to the system to return all and only the patterns which exhibit an interest level larger than a given threshold λ.
- The proposed paradigm allows to assign different weights to different constraints, while in the classical constraint-based pattern discovery paradigm all constraints were equally important.
- Last but not least, our idea is very simple and thus very general: it can be instantiated to different classes of patterns such as itemsets, sequences, trees or graphs.

For the reasons listed above, we believe that the proposed paradigm represents an important step further towards practical pattern discovery.

A nice feature of our proposal is that, by adopting the soft constraint based paradigm, we do not reject all research results obtained in the classical constraint-based paradigm; on the contrary, we fully exploit such algorithmic results. In other terms, our proposal is merely methodological, and it exploits previous research results that were mainly computational.

The paper is organized as follows. In the next Section we briefly review the theory of soft constraints and we define the soft constraint based pattern mining paradigm. In Section 3 we discuss possible alternative instances of the paradigm. In Section 4 we formally define the Soft Constraint Based Pattern Discovery paradigm. We then focus on one of the many possible instances of the proposed paradigm, and we implement it in a concrete Pattern Discovery System. Such a system is built as a wrapper around a classical constraint pattern mining system.

2 Introducing Soft Constraints

Constraint Solving is an emerging software technology for declarative description and effective solving of large problems. Many real life systems, ranging from network management [14] to complex scheduling [2], are analyzed and solved using constraint related technologies. The constraint programming process consists of the generation of requirements (constraints) and solution of these requirements, by specialized constraint solvers. When the requirements of a problem are expressed as a collection of boolean predicates over variables, we obtain what is called the *crisp* (or classical) Constraint Satisfaction Problem (CSP). In this case the problem is solved by finding any assignment of the variables that satisfies all the constraints.

Sometimes, when a deeper analysis of a problem is required, *soft constraints* are used instead. Several formalizations of the concept of soft constraints are currently available. In the following, we refer to the formalization based on *c-semirings* [5]: a semiring-based constraint assigns to each instantiation of its variables an associated value from a partially ordered set. When dealing with crisp constraints, the values are the boolean $true$ and $false$ representing the admissible and/or non-admissible values; when dealing with soft constraints the values are interpreted as preferences/costs. The framework must also handle the combination of constraints. To do this one must take into account such additional values, and thus the formalism must provide suitable operations for combination ($\times$) and comparison ($+$) of tuples of values and constraints. This is why this formalization is based on the mathematical concept of semiring.

Definition 3 (c-semirings [5, 3]). *A semiring is a tuple $\langle A, +, \times, \mathbf{0}, \mathbf{1}\rangle$ such that: A is a set and $\mathbf{0}, \mathbf{1} \in A$; $+$ is commutative, associative and $\mathbf{0}$ is its unit element; $\times$ is associative, distributes over $+$, $\mathbf{1}$ is its unit element and $\mathbf{0}$ is its absorbing element. A c-semiring ("c" stands for "constraint-based") is a semiring $\langle A, +, \times, \mathbf{0}, \mathbf{1}\rangle$ such that $+$ is idempotent with $\mathbf{1}$ as its absorbing element and $\times$ is commutative.*

Definition 4 (soft constraints [5, 3]). *Given a c-semiring $S = \langle A, +, \times, \mathbf{0}, \mathbf{1}\rangle$ and an ordered set of variables V over a finite domain D, a constraint is a function which, given an assignment $\eta : V \rightarrow D$ of the variables, returns a value of the c-semiring. By using this notation we define $\mathcal{C} = \eta \rightarrow A$ as the set of all possible constraints that can be built starting from S, D and V.*

In the following we will always use the word semiring as standing for c-semiring, and we will explain this very general concept by the point of view of pattern discovery.

Example 2. Consider again the mining query $\mathcal{Q}$. In this context we have that the ordered set of variables V is $\langle supp_{\mathcal{D}}(X), avg(X.weight), sum(X.price)\rangle$, while the domain D is: $D(supp_{\mathcal{D}}(X)) = \mathbb{N}$, $D(avg(X.weight)) = \mathbb{R}^+$, and $D(sum(X.price)) = \mathbb{N}$. If we consider the classical *crisp* framework (i.e., hard constraints) we have the semiring $S_{Bool} = \langle \{true, false\}, \vee, \wedge, false, true\rangle$. A constraint C is a function $V \rightarrow D \rightarrow A$; for instance, $supp_{\mathcal{D}}(X) \rightarrow 1700 \rightarrow true$.

The $+$ operator is what we use to compare tuples of values (or patterns, in our context). Let us consider the relation $\leq_S$ (where S stands for the specified semiring) over

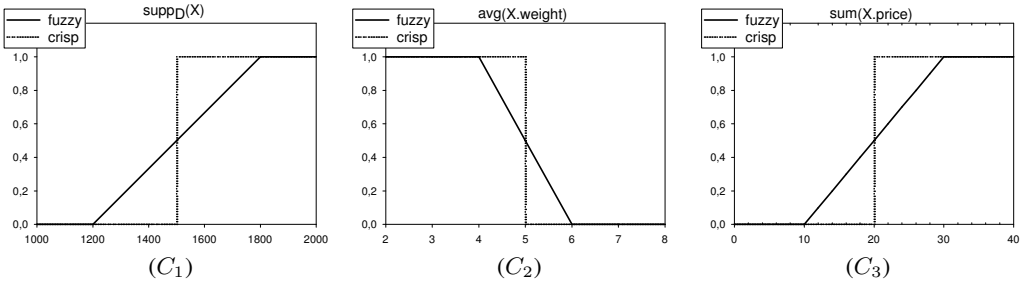
A such that $a \leq_S b$ iff $a + b = b$. It is possible to prove that: $\leq_S$ is a partial order; $+$ and $\times$ are monotone on $\leq_S$; $\mathbf{0}$ is its minimum and $\mathbf{1}$ its maximum, and $\langle A, \leq_S \rangle$ is a complete lattice with least upper bound operator $+$. In the context of pattern discovery $a \leq_S b$ means that the pattern b is *more interesting* than a, where interestingness is defined by a combination of soft constraints. When using (soft) constraints it is necessary to specify, via suitable combination operators, how the level of interest of a combination of constraints is obtained from the interest level of each constraint. The combined weight (or interest) of a combination of constraints is computed by using the operator $\otimes : \mathcal{C} \times \mathcal{C} \rightarrow \mathcal{C}$ defined as $(C_1 \otimes C_2)\eta = C_1\eta \times_S C_2\eta$.

Example 3. If we adopt the classical crisp framework, in the mining query $\mathcal{Q}$ of Example 1 we have to combine the three constraints using the $\wedge$ operator (which is the $\times$ in the boolean semiring S_{Bool}). Consider for instance the pattern $p_1 : \langle 1700, 0.8, 19 \rangle$ for the ordered set of variables $V = \langle supp_{\mathcal{D}}(X), avg(X.weight), sum(X.price) \rangle$. The first and the second constraint are satisfied leading to the semiring level $true$, while the third one is not satisfied and has associated level $false$. Combining the three values with $\wedge$ we obtain $true \wedge true \wedge false = false$ and we can conclude that the pattern $\langle 1700, 0.8, 19 \rangle$ is not interesting w.r.t. our purposes. Similarly, we can instead compute level $true$ for pattern $p_3 : \langle 1550, 2.2, 26 \rangle$ corresponding to an interest w.r.t. our goals. Notice that using crisp constraints, the order between values only says that we are interested to patterns with semiring level $true$ and not interested to patterns with semiring level $false$ (that is semiring level $false \leq_{S_{Bool}} true$).

3 Instances of the Semiring

Dividing patterns in *interesting* and *non-interesting* is sometimes not meaningful nor useful. Most of the times we can say that each pattern is interesting with a specific level of preference. Soft constraints can deal with preferences by moving from the two values semiring S_{Bool} to other semirings able to give a finer distinction among patters (see [3] for a comprehensive guide to the semiring framework). For our scope the fuzzy and the weighted semiring are the most suitable.

Example 4 (fuzzy semiring). When using fuzzy semiring [12, 23], to each pair constraint-pattern is assigned an interest level between 0 and 1, where 1 represents the best value (maximum interest) and 0 the worst one (minimum interest). Therefore the $+$ in this semiring is given by the max operator, and the order $\leq_S$ is given by the usual $\leq$ on real numbers. The value associated to a pattern is obtained by combining the constraints using the minimum operator among the semiring values. Therefore the $\times$ in this semiring is given by the min operator. Recapitulating, the fuzzy semiring is given by $S_F = \langle [0, 1], max, min, 0, 1 \rangle$. The reason for such a max-min framework relies on the attempt to maximize the value of the least preferred tuple. Fuzzy soft constraints are able to model partial constraint satisfaction [13], so to get a solution even when the problem is overconstrained, and also prioritized constraints, that is, constraints with different levels of importance [10]. Figure 1 reports graphical representations of possible fuzzy instances of the constraints in $\mathcal{Q}$. Consider, for instance, the graphical representation of the frequency constraint in Figure 1(C_1). The dotted line describes the behavior



(C_1) (C_2) (C_3)

Fig. 1. Graphical representation of possible fuzzy instances of the constraints in $\mathcal{Q}$.

of the *crisp* version (where $1 = true$ and $0 = false$) of the frequency constraint, while the solid line describes a possible fuzzy instance of the same constraint. In this instance domain values smaller than 1200 yield 0 (uninteresting patterns); from 1200 to 1800 the interest level grows linearly reaching the maximum value of 1. Similarly the other two constraints in Figure 1(C_2) and (C_3). In this situation for the pattern $p_1 = \langle 1700, 0.8, 19\rangle$ we obtain that: $C_1(p_1) = 1$, $C_2(p_1) = 1$ and $C_3(p_1) = 0.45$. Since in the fuzzy semiring the combination operator $\times$ is min, we got that the interest level of p_1 is 0.45. Similarly for p_2 and p_3:

- $p_1 : C_1 \otimes C_2 \otimes C_3(1700, 0.8, 19) = min(1, 1, 0.45) = 0.45$
- $p_2 : C_1 \otimes C_2 \otimes C_3(1550, 4.8, 54) = min(1, 0.6, 1) = 0.6$
- $p_3 : C_1 \otimes C_2 \otimes C_3(1550, 2.2, 26) = min(1, 1, 0.8) = 0.8$

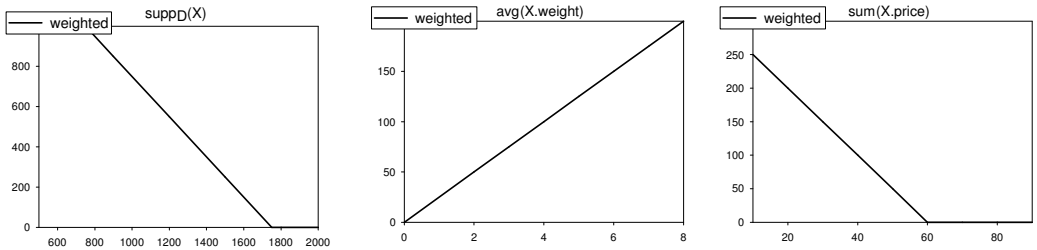
Therefore, with this particular instance we got that $p_1 \leq_{S_F} p_2 \leq_{S_F} p_3$, i.e., p_3 is the most interesting pattern among the three.

Example 5 (weighted semiring). While fuzzy semiring associate a level of preference with each tuple in each constraint, in the weighted semiring tuples come with an associated cost. This allows one to model optimization problems where the goal is to minimize the total cost (time, space, number of resources, ...) of the proposed solution. Therefore, in the weighted semiring the cost function is defined by summing up the costs of all constraints. According to the informal description given above, the weighted semiring is $S_W = \langle \mathbb{R}^+, min, sum, +\infty, 0\rangle$. Consider, for instance, the graphical representation of the constraints in the query $\mathcal{Q}$ in Figure 2. In this situation we got that:

- $p_1 : C_1 \otimes C_2 \otimes C_3(1700, 0.8, 19) = sum(50, 20, 205) = 275$
- $p_2 : C_1 \otimes C_2 \otimes C_3(1550, 4.8, 54) = sum(200, 120, 30) = 350$
- $p_3 : C_1 \otimes C_2 \otimes C_3(1550, 2.2, 26) = sum(200, 55, 190) = 445$

Therefore, with this particular instance we got that $p_3 \leq_{S_W} p_2 \leq_{S_W} p_1$ (remember that the order $\leq_{S_W}$ correspond to the $\geq$ on real numbers). In other terms, p_1 is the most interesting pattern w.r.t. this constraints instance.

The weighted and the fuzzy paradigm, can be seen as two different approaches to give a meaning to the notion of optimization. The two models correspond in fact to two definitions of social welfare in utility theory [19]: *"egalitarianism"*, which maximizes the minimal individual utility, and *"utilitarianism"*, which maximizes the sum of the



(C_1) (C_2) (C_3)

Fig. 2. Graphical representation of possible weighted instances of the constraints in $\mathcal{Q}$.

individual utilities. The fuzzy paradigm has an egalitarianistic approach, aimed at maximizing the overall level of interest while balancing the levels of all constraints; while the weighted paradigm has an utilitarianistic approach, aimed at getting the minimum cost globally, even though some constraints may be neglected presenting a big cost. We believe that both approaches present advantages and drawbacks, and may preferred to the other one depending on the application domain. Beyond the fuzzy and the weighted, many other possible instances of the semiring exist, and could be useful in particular applications. Moreover, it is worth noting that the cartesian product of semirings is a semiring [5] and thus it is possible to use the framework also to deal with multicriteria pattern selection.

Finally, note that the soft constraint framework is very general, and could be instantiated not only to unary constraints (as we do in this paper) but also to binary and k-ary constraints (dealing with two or more variables). This could be useful to extend the soft constraint based paradigm to association rules with "2-var" constraints [18].

4 Soft Constraint Based Pattern Mining

In this Section we instantiate soft constraint theory to the pattern discovery framework.

Definition 5 (Soft Constraint Based Pattern Mining). *Let $\mathcal{P}$ denote the domain of possible patterns. A soft constraint on patterns is a function $\mathcal{C} : \mathcal{P} \rightarrow A$ where A is the carrier set of a semiring $S = \langle A, +, \times, \mathbf{0}, \mathbf{1} \rangle$. Given a combination of soft constraints $\otimes\mathcal{C}$, we define two different problems:*

λ-interesting: *given a minimum interest threshold $\lambda \in A$, it is required to mine the set of all λ-interesting patterns, i.e., $\{p \in \mathcal{P} | \otimes \mathcal{C}(p) \geq \lambda\}$.*

top-k: *given a threshold $k \in \mathbb{N}$, it is required to mine the top-k patterns $p \in \mathcal{P}$ w.r.t. the order $\leq_S$.*

Note that the Soft Constraint Based Pattern Mining paradigm just defined, has many degrees of freedom. In particular, it can be instantiated: (i) on the domain of patterns $\mathcal{P}$ in analysis (e.g., itemsets, sequences, trees or graphs), (ii) on the semiring $S = \langle A, +, \times, \mathbf{0}, \mathbf{1} \rangle$ (e.g., fuzzy, weighted or probabilistic), and (iii) on one of the two possible mining problems, i.e., λ-interesting or top-k mining.

In the rest of this paper we will focus on concretizing a simple instance of this very general paradigm: λ-interesting$_{fuzzy}$ on the pattern class of itemsets.

4.1 Mining λ-interesting Itemsets on the Fuzzy Semiring

Definition 6. *Let $\mathcal{I} = \{x_1, ..., x_n\}$ be a set of items, where an item is an object with some predefined attributes (e.g., price, type, etc.). A soft constraint on itemsets, based on the fuzzy semiring, is a function $\mathcal{C} : 2^{\mathcal{I}} \rightarrow [0, 1]$. Given a combination of such soft constraints $\otimes\mathcal{C} \equiv \mathcal{C}_1 \otimes \ldots \otimes \mathcal{C}_n$, we define the interest level of an itemset $X \in 2^{\mathcal{I}}$ as $\otimes\mathcal{C}(X) = min(\mathcal{C}_1(X), \ldots, \mathcal{C}_n(X))$. Given a minimum interest threshold $\lambda \in]0, 1]$, the λ-interesting itemsets mining problem, requires to compute $\{X \in 2^{\mathcal{I}} | \otimes\mathcal{C}(X) \geq \lambda\}$.*

In the following we describe how to build a concrete *pattern discovery system* for λ-interesting$_{fuzzy}$ itemsets mining, as a wrapper around a classical constraint pattern mining system. The basic components which we use to build our system are the following:

A *crisp* constraints solver - i.e., a system for mining constrained frequent itemsets, where constraints are classical binary functions, and not soft constraints. Or in other terms, a system for solving the problem in Definition 2. To this purpose we adopt the system which we have developed at Pisa KDD Laboratory within the P^3D project[1]. Such a system is a general Apriori-like algorithm which, by means of *data reduction* and *search space pruning*, is able to push a wide variety of constraints (practically all possible kinds of constraints which have been studied and characterized so far [9]) into the frequent itemsets computation. Based on the algorithmic results developed in the last years by our lab (e.g., [6–9, 21]), our system is very efficient and robust, and to our knowledge, is the unique existing implementation of this kind.

A language of constraints - to express, by means of queries containing conjunctions of constraints, what is interesting for the given application. The wide repertoire of constraints that we admit, comprehends the frequency constraint ($supp_{\mathcal{D}}(X) \geq \sigma$), and all constraints defined over the following aggregates[2]: $min, max, count, sum, range, avg, var, median, std, md$.

A methodology to define the interest level - that must be assigned to each pair itemset-constraint. In other terms, we need to provide the analyst with a simple methodology to define how to assign for each constraint and each itemset a value in the interval $[0, 1]$, as done, for instance, by the graphical representations of constraints in Figure 1. This methodology should provide the analyst with a knob to adjust the *softness level* of each constraint in the conjunction, and a knob to set the *importance* of each constraint in the conjunction.

Let us focus on the last point. Essentially we must describe how the user can define the fuzzy behavior of a soft constraint. We restrict our system to constraints which behave as those ones in Figure 1: they return a value which grows linearly from 0 to 1 in a certain interval, while they are null before the interval and equal to 1 after the interval. To describe such a simple behavior we just need two parameters: a value associated to the center of the interval (corresponding to the 0.5 fuzzy semiring value), and a parameter to adjust the width of the interval (and consequently the gradient of the function).

[1] `http://www-kdd.isti.cnr.it/p3d/index.html`

[2] $range$ is $(max - min)$, var is for variance, std is for standard deviation, md is for mean deviation.

Definition 7. *A soft constraint $\mathcal{C}$ on itemsets, based on the fuzzy semiring, is defined by a quintuple $\langle Agg, Att, \theta, t, \alpha \rangle$, where:*

- $Agg \in \{supp, min, max, count, sum, range, avg, var, median, std, md\}$*;*
- Att *is the name of the attribute on which the aggregate agg is computed (or the transaction database, in the case of the frequency constraint);*
- $\theta \in \{\leq, \geq\}$*;*
- $t \in \mathbb{R}$ *corresponds to the center of the interval and it is associated to the semiring value 0.5;*
- $\alpha \in \mathbb{R}^+$ *is the softness parameter, which defines the inclination of the preference function (and thus the width of the interval).*

In particular, if $\theta = \leq$ (as in Figure $1(C_2)$) then $\mathcal{C}(X)$ is 1 for $X \leq (t - \alpha t)$, is 0 for $X \geq (t + \alpha t)$, and is linearly decreasing from 1 to 0 within the interval $[t - \alpha t, t + \alpha t]$. The other way around if $\theta = \geq$ (as, for instance, in Figure $1(C_3)$). Note that if the softness parameter α is 0, then we obtain the crisp (or hard) version of the constraint.

Example 6. Consider again the query $\mathcal{Q}$ given in Example 1, and its fuzzy instance graphically described by Figure 1. Such query can be expressed in our constraint language as: $\langle supp, \mathcal{D}, \geq, 1500, 0.2 \rangle, \langle avg, weight, \leq, 5, 0.2 \rangle, \langle sum, price, \geq, 20, 0.5 \rangle$.

Since the combination operator $\times$ in min, increasing the importance of a constraint w.r.t. the others in the combination means to force the constraint to return lower values for not really satisfactory patterns. By decreasing the softness parameter α, we increase the gradient of the function making the shape of the soft constraint closer to a crisp constraint. This translates in a better value for patterns X which were already behaving well w.r.t. such constraint($\mathcal{C}(X) > 0.5$), and in a lower value for patterns which were behaving not so well ($\mathcal{C}(X) < 0.5$). Decreasing the gradient (increasing α) instead means to lower the importance of the constraint itself: satisfying or not satisfying the constraint does not result in a big fuzzy value difference. Additionally, by operating on t, we can increase the "severity" of the constraint w.r.t. those patterns which were behaving not so well. Therefore, the knob to increase or decrease the importance of a constraint is not explicitly given, because its role, in the fuzzy semiring, can be played by a combined action on the two knobs α and t.

Example 7. Consider again the query $\mathcal{Q}$ given in Example 1, and its fuzzy instance: $\langle supp, \mathcal{D}, \geq, 1500, 0.2 \rangle, \langle avg, weight, \leq, 5, 0.2 \rangle, \langle sum, price, \geq, 20, 0.5 \rangle$. As we stated in Example 4, it holds that $p_2 \leq_{S_F} p_3$. In particular, p_2 is better than p_3 w.r.t. constraint C_3, while p_3 is better than p_2 w.r.t. constraint C_2. Suppose now that we increase the importance of C_3, e.g., $\langle sum, price, \geq, 28, 0.25 \rangle$. We obtain that $p_3 \leq_{S_F} p_2$:

- $p_2 : C_1 \otimes C_2 \otimes C_3(1550, 4.8, 54) = min(1, 0.6, 1) = 0.6$
- $p_3 : C_1 \otimes C_2 \otimes C_3(1550, 2.2, 26) = min(1, 1, 0.35) = 0.35$

In [5, 4] it has been proved that, when dealing with the fuzzy framework, computing all the solution better than a threshold λ can be performed by solving a crisp problem where all the constraint instances with semiring level lower than λ have been assigned level $false$, and all the instances with semiring level greater or equal to λ have been assigned level $true$. Using this theoretical result, and some simple arithmetic we can transform each soft constraint in a corresponding crisp constraint.

Definition 8. *Given a fuzzy soft constraint $\mathcal{C} \equiv \langle Agg, Att, \theta, t, \alpha \rangle$, and a minimum interest threshold λ, we define the crisp translation of $\mathcal{C}$ w.r.t. λ as:*

$$\mathcal{C}^{\lambda}_{crisp} \equiv \begin{cases} Agg(Att) \geq t - \alpha t + 2\lambda\alpha t, & \text{if } \theta = \geq \\ Agg(Att) \leq t + \alpha t - 2\lambda\alpha t, & \text{if } \theta = \leq \end{cases}$$

Example 8. The crisp translation of the soft constraint $\langle sum, price, \geq, 20, 0.5 \rangle$ is $sum(X.price) \geq 26$ for $\lambda = 0.8$, while it is $sum(X.price) \geq 18$ for $\lambda = 0.4$.

Proposition 1. *Given the vocabulary of items $\mathcal{I}$, a combination of soft constraints $\otimes\mathcal{C} \equiv \mathcal{C}1 \otimes \ldots \otimes \mathcal{C}n$, and a minimum interest threshold λ. Let $\mathcal{C}'$ be the conjunction of crisp constraints obtained by conjoining the crisp translation of each constraint in $\otimes\mathcal{C}$ w.r.t. λ: $\mathcal{C}' \equiv \mathcal{C}1^{\lambda}_{crisp} \wedge \ldots \wedge \mathcal{C}n^{\lambda}_{crisp}$. It holds that: $\{X \in 2^{\mathcal{I}} | \otimes\mathcal{C}(X) \geq \lambda\} = Th(\mathcal{C}')$.*

Proof (sketch). The soundness of the mapping come from the result in [5]. We here have to only give a justification of the formula in Definition 8. This is done by means of Figure 3(b), that shows a graphical representation of the simple arithmetic problem and its solutions.

Therefore, if we adopt the fuzzy semiring, we can fully exploit a classical constraint-based pattern discovery system (and all algorithmic results behind it), by means of a simple translation from soft to crisp constraints. This is exactly what we have done, obtaining a pattern discovery system based on soft constraints built as a wrapper around a classical constraint-based mining system.

4.2 Experimental Analysis

We have conducted some experiments in order to asses the concrete effects obtained by manipulating the α, t and λ parameters. To this purpose we have compared 5 different instances (described in Figure 3(a)) of the query $\mathcal{Q}$:

$$\langle supp, \mathcal{D}, \geq, t, \alpha \rangle \langle avg, weight, \leq, t, \alpha \rangle, \langle sum, price, \geq, t, \alpha \rangle$$

where the transactional dataset $\mathcal{D}$, is the well known RETAIL dataset, donated by Tom Brijs and contains the (anonymized) retail market basket data from an anonymous Belgian retail store[3]; and the two attributes $weight$ and $price$ have been randomly generated with a gaussian distribution within the range $[0, 150000]$.

Figure 3(c) reports the number of solutions for the given five queries at different λ thresholds. Obviously as λ increases the number of solutions shrinks accordingly. This behavior is also reflected in queries evaluation times, reported in Figure 3(d): the bigger is the size of the solution set, the longer is the associated computation.

Comparing queries $\mathcal{Q}_1$, $\mathcal{Q}_2$ and $\mathcal{Q}_3$, we can gain more insight about the α parameter. In fact, the three queries differ only by the α associated with one constraint (the frequency constraint). We can observe that, if the λ threshold is not too much selective, increasing the α parameter (i.e., the size of the soft interval), the number of solutions

[3] `http://fimi.cs.helsinki.fi/data/`

(a)

	$\langle supp, \mathcal{D}, \geq, t, \alpha \rangle$		$\langle avg, weight, \leq, t, \alpha \rangle$		$\langle sum, price, \geq, t, \alpha \rangle$	
	t	α	t	α	t	α
$\mathcal{Q}_1$	20	0.8	10000	0.5	20000	0.5
$\mathcal{Q}_2$	20	0.5	10000	0.5	20000	0.5
$\mathcal{Q}_3$	20	0.2	10000	0.5	20000	0.5
$\mathcal{Q}_4$	20	0.8	5000	0.2	20000	0.5
$\mathcal{Q}_5$	20	0.8	5000	0.8	20000	0.5

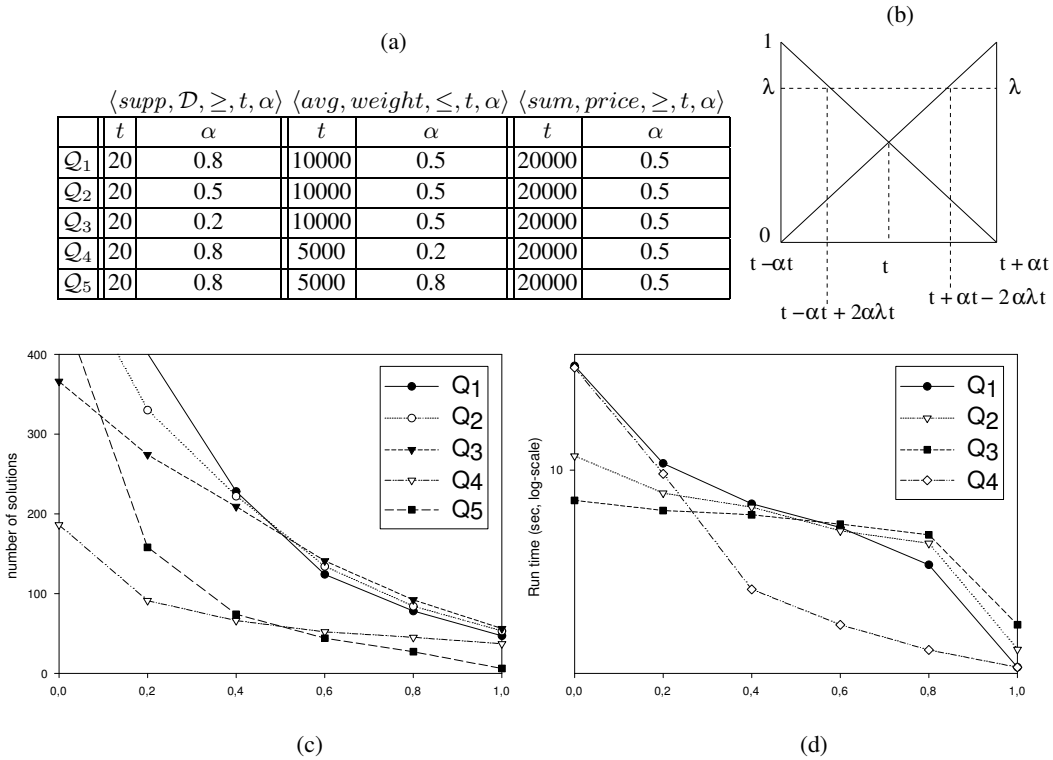


Fig. 3. (a) description of queries experimented, (b) graphical proof to Proposition 1, (c) and (d) experimental results with λ ranging in $]0, 1]$.

grows. Notice however that, when λ becomes selective enough (i.e., $\lambda > 0.5$), increasing the softness parameter we obtain an opposite behavior. This is due to the fact that, if on one hand a more soft constraint is less severe with patterns not good enough, on the other hand it is less generous with good patterns, which risk to be discarded by an high λ threshold.

References

1. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases. In *Proc. of the 20th International Conference on Very Large Databases (VLDB'94)*.
2. J. Bellone, A. Chamard, and C. Pradelles. Plane - an evolutive planning system for aircraft production. In *Proc. of the 1st Interantional Conference on Practical Applications of Prolog (PAP'92)*.
3. S. Bistarelli. *Semirings for Soft Constraint Solving and Programming*, volume 2962 of *Lecture Notes in Computer Science*. Springer, 2004.
4. S. Bistarelli, P. Codognet, and F. Rossi. Abstracting soft constraints: Framework, properties, examples. *Artificial Intelligence*, (139):175–211, July 2002.
5. S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based Constraint Solving and Optimization. *Journal of the ACM*, 44(2):201–236, Mar 1997.

6. F. Bonchi, F. Giannotti, A. Mazzanti, and D. Pedreschi. ExAMiner: Optimized level-wise frequent pattern mining with monotone constraints. In *Proc. of the 3rd IEEE International Conference on Data Mining (ICDM'03)*.
7. F. Bonchi, F. Giannotti, A. Mazzanti, and D. Pedreschi. ExAnte: Anticipated data reduction in constrained pattern mining. In *Proc. of the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'03)*.
8. F. Bonchi and C. Lucchese. On closed constrained frequent pattern mining. In *Proc. of the 4th IEEE International Conference on Data Mining (ICDM'04)*.
9. F. Bonchi and C. Lucchese. Pushing tougher constraints in frequent pattern mining. In *Proc. of the 9th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'05)*.
10. A. Borning, M. Maher, A. Martindale, and M. Wilson. Constraint hierarchies and logic programming. In *Proc. 6th International Conference on Logic Programming*, 1989.
11. C. Bucila, J. Gehrke, D. Kifer, and W. White. DualMiner: A dual-pruning algorithm for itemsets with constraints. In *Proc. of the 8th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'02)*.
12. D. Dubois, H. Fargier, and H. Prade. The calculus of fuzzy restrictions as a basis for flexible constraint satisfaction. In *Proc. of the 2nd IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'93)*.
13. E. Freuder and R. Wallace. Partial constraint satisfaction. *AI Journal*, 58, 1992.
14. T. Frühwirth and P. Brisset. Optimal planning of digital cordless telecommunication systems. In *Proc. of the 3rd International Conference on The Practical Application of Constraint Technology (PACT'97)*.
15. J. Han, L. V. S. Lakshmanan, and R. T. Ng. Constraint-based, multidimensional data mining. *Computer*, 32(8):46–50, 1999.
16. R. Hilderman and H. Hamilton. *Knowledge Discovery and Measures of Interest*. Kluwer Academic, Boston, 2002.
17. S. Kramer, L. D. Raedt, and C. Helma. Molecular feature mining in hiv data. In *Proc. of the 7th ACM International Conference on Knowledge Discovery and Data Mining, (SIGKDD'01)*.
18. L. V. S. Lakshmanan, R. T. Ng, J. Han, and A. Pang. Optimization of constrained frequent set queries with 2-variable constraints. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD'99)*.
19. H. Moulin. *Axioms for Cooperative Decision Making*. Cambridge University Press, 1988.
20. R. T. Ng, L. V. S. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD'98)*.
21. S. Orlando, P. Palmerini, R. Perego, and F. Silvestri. Adaptive and Resource-Aware Mining of Frequent Sets. In *Proc. of the 2nd IEEE Int. Conference on Data Mining (ICDM'02)*.
22. J. Pei and J. Han. Can we push more constraints into frequent pattern mining? In *Proceedings of the 6th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'00)*.
23. Z. Ruttkay. Fuzzy constraint satisfaction. In *Proc. 3rd IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'94)*.
24. S. Sahar. Interestingness via what is not interesting. In *Proc. of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'99)*.
25. R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In *Proc. of the 3rd ACM International Conference on Knowledge Discovery and Data Mining, (SIGKDD'97)*.
26. P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'02)*.