Mining Implications from Lattices of Closed Trees

José L. Balcázar*, Albert Bifet*, Antoni Lozano *

* Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
{balqui,abifet,antoni}@lsi.upc.edu

Abstract. We propose a way of extracting high-confidence association rules from datasets consisting of unlabeled trees. The antecedents are obtained through a computation akin to a hypergraph transversal, whereas the consequents follow from an application of the closure operators on unlabeled trees developed in previous recent works of the authors. We discuss in more detail the case of rules that always hold, independently of the dataset, since these are more complex than in itemsets due to the fact that we are no longer working on a lattice.

1 Introduction

In the field of data mining, one of the major notions contributing to the success of the area has been that of association rules. Many studies of various types have provided a great advance of the human knowledge about these concepts. One particular family of studies is rooted on the previous notions of formal concepts, Galois lattices, and implications, which correspond to association rules of maximum confidence.

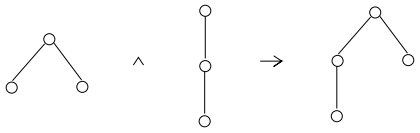
These notions have allowed for more efficient works and algorithmics by reducing the computation of frequent sets, a major usual step towards association rules, to the computation of so-called closed frequent sets, a faster computation of much more manageable output size, yet losing no information at all with respect to frequent sets.

It was realized some time ago that the plain single-relational model for the data, as employed by the computation of either closed sets or association rules, whereas useful to a certain extent, was a bit limited in its applicability by the fact that, often, real-life data have some sort of internal structure that is lost in the transactional framework. Thus, studies of data mining in combinatorial structures were undertaken, and considerable progress has been made in recent years. Our work here is framed in that endeavor.

In previous work, we have proposed a mathematical clarification of the closure operator underlying the notion of closed trees in datasets of trees; the closure operator no longer works on single trees but on sets of them. In a sense, made precise there, closed trees do not constitute a lattice. A mathematically precise replacement lattice can be defined, though, as demonstrated in (Balcázar et al., 2006), consisting not anymore of trees but of sets of trees, and with the peculiar property that, in all experiments with real-life data we have undertaken, they turn out to be actually lattices of trees, in the sense that every closed set of trees was, in all practical cases, a singleton.

Algorithmics to construct these closed sets have been studied in several references as (Chi et al., 2005b), (Balcázar et al., 2007a), (Balcázar et al., 2007b), (Termier et al., 2004); see the references in the survey (Chi et al., 2005a). We continue here this line of research by tackling the most natural next step: the identification of implications out of the lattice of closed sets of trees. We describe a method, along the line of similar works on sequences and partial orders (Balcázar and Garriga (2007b), Balcázar and Garriga (2007a)) to construct implications from the closed sets of trees, and we mathematically characterize, in terms of propositional Horn theories, the implications that we find.

Then, we explain a major difference of our case with previous works: rules that would not be trivial in other cases become redundant, and thus unnecessary, in the case of trees, due to the fact that they are implicit in the combinatorics of the structures. An example will show best our point here; please bear for now with some undefined notions that will be precised later on. Consider a rule intuitively depicted as follows:



It naturally means that whenever a tree in the dataset under exploration has as (top-down) subtrees the two trees in the antecedent, it also has the one in the consequent. Any tree having a bifurcation at the root, as required by the first antecedent, and a branch of length at least two, as required by the second one, has to have the consequent as a (top-down) subtree. Therefore, the rule says, in fact, nothing at all about the dataset, and is not worthy to appear in the output of a rule mining algorithm on trees.

Our second major contribution is, therefore, a study of some cases where we can detect such implicit rules and remove them from the output, with low computational overhead. Whereas further theoretical work might be useful, our contributions so far already detect most of the implicit rules in real-life datasets, up to quite low support levels, and with a reasonable efficiency. We report some facts on the empirical behavior of our implementations of both the algorithm to find rules and the heuristics to remove implicit rules.

2 Preliminaries

Trees are connected acyclic graphs, *rooted trees* are trees with a vertex singled out as the root, and *unranked* trees are trees with unbounded arity. We say that $t_1, \ldots, t_k$ are the *components* of tree t if t is made of a node (the root) joined to the roots of all the t_i's. We can distinguish between the cases where the components at each node form a sequence (ordered trees) or just a set (*unordered trees*). We will deal with rooted, unranked, unordered trees. We do not assume the presence of labels on the nodes. The (infinite) set of all trees will be denoted with $\mathcal{T}$, but actually all our developments will proceed in some finite subset of $\mathcal{T}$ which will act as our universe of discourse.

In order to compare link-based structures, we are interested in a notion of subtree where the root is preserved. A tree t' is a *top-down subtree* (or simply a *subtree*) of a tree t (written $t' \preceq t$) if t' is a connected subgraph of t which contains the root of t. This notation can be extended to sets of trees $A \preceq B$: for all $t \in A$, there is some $t' \in B$ for which $t \preceq t'$. Two

trees t, t' are said to be *comparable* if $t \preceq t'$ or $t' \preceq t$. Otherwise, they are incomparable. Also $t \prec t'$ if t is a proper subtree of t' (that is, $t \preceq t'$ and $t \neq t'$).

The input to our data mining process is a given finite dataset $\mathcal{D}$ of transactions, where each transaction $s \in \mathcal{D}$ consists of a transaction identifier, tid, and an unlabeled rooted tree. Tids are supposed to run sequentially from 1 to the size of $\mathcal{D}$. From that dataset, our universe of discourse $\mathcal{U}$ is the set of all trees that appear as subtree of some tree in $\mathcal{D}$.

Following standard usage, we say that a transaction s *supports* a tree t if t is a subtree of the tree corresponding to transaction s. The number of transactions in the dataset $\mathcal{D}$ that supports t is called the *support* of the tree t. A subtree t is called *frequent* if its support is greater than or equal to a given threshold min_sup. The frequent subtree mining problem is to find all frequent subtrees in a given dataset. Any subtree of a frequent tree is also frequent and, therefore, any supertree of a nonfrequent tree is also nonfrequent.

We define a frequent tree t to be *closed* if none of its proper supertrees has the same support as it has. Generally, there are much fewer closed trees than frequent ones. In fact, we can obtain all frequent subtrees with their support from the set of closed frequent subtrees with their supports.

The dataset defines a closure operator on the powerset of $\mathcal{U}$, denoted $\Gamma_{\mathcal{D}}$, arising from a Galois connection and developed in (Balcázar et al., 2006).

Definition The Galois connection pair:

- For finite $A \subseteq \mathcal{D}$, $\sigma(A) = \{t \in \mathcal{T} \mid \forall\, t' \in A\, (t \preceq t')\}$
- For finite $B \subset \mathcal{T}$, not necessarily in $\mathcal{D}$, $\tau_{\mathcal{D}}(B) = \{t' \in \mathcal{D} \mid \forall\, t \in B\, (t \preceq t')\}$

Proposition 2.1 *The composition $\Gamma_{\mathcal{D}} = \sigma \circ \tau_{\mathcal{D}}$ is a closure operator.*

Theorem 2.2 *A tree t is closed for $\mathcal{D}$ if and only if it is maximal in $\Gamma_{\mathcal{D}}(\{t\})$.*

We will construct association rules in a standard form from it, and show that they correspond to a certain Horn theory; also, we will prove the correctness of a construction akin to the instantaneous basis of Wild (1994) and Pfaltz and Taylor (2002).

3 Association Rules

Following standard usage on Galois lattices, we consider now implications (sometimes called deterministic association rules, see e.g. Pfaltz and Taylor (2002)) of the form $A \to B$ for sets of trees A and B from $\mathcal{U}$. Specifically, we consider the following set of rules: $A \to \Gamma_{\mathcal{D}}(A)$. Alternatively, we can split the consequents into $\{A \to t \mid t \in \Gamma_{\mathcal{D}}(A)\}$.

It is easy to see that $\mathcal{D}$ obeys all these rules: for each A, any tree of $\mathcal{D}$ that has as subtrees all the trees of A has also as subtrees all the trees of $\Gamma_{\mathcal{D}}(A)$.

We want to provide a characterization of this set of implications. We operate in a form similar to Balcázar and Garriga (2007a) and Balcázar and Garriga (2007b), translating this set of rules into a specific propositional theory which we can characterize, and for which we can find a "basis": a set of rules that are sufficient to infer all the rules that hold in the dataset $\mathcal{D}$. The technical details depart somewhat from Balcázar and Garriga (2007b) in that we skip a

certain maximality condition imposed there, and are even more different from those in Balcázar and Garriga (2007a).

Thus, we start by associating a propositional variable v_t to each tree $t \in \mathcal{U}$. In this way, each implication between sets of trees can be seen also as a propositional conjunction of Horn implications, as follows: the conjunction of all the variables corresponding to the set at the left hand side implies each of the variables corresponding to the closure at the right hand side. We call this propositional Horn implication the propositional translation of the rule.

Also, now a set of trees A corresponds in a natural way to a propositional model m_A: specifically, $m_A(v_t) = 1$ if and only if t is a subtree of some tree in A. We abbreviate $m_{\{t\}}$ as m_t. Note that the models obtained in this way obey the following condition: if $t' \preceq t$ and $v_t = 1$, then $v_{t'} = 1$ too. In fact, this condition identifies the models m_A: if a model m fulfills it, then $m = m_A$ for the set A of trees t for which $v_t = 1$ in m. Alternatively, A can be taken to be the set of maximal trees for which $v_t = 1$.

Note that we can express this condition by a set of Horn clauses: $\mathcal{R}_0 = \{v_{t'} \rightarrow v_t \mid t' \preceq t,\ t \in \mathcal{U},\ t' \in \mathcal{U}\}$. It is easy to see that the following holds:

Lemma 3.1 *Let $t \in \mathcal{D}$. Then m_t satisfies $\mathcal{R}_0$ and also all the propositional translations of the implications of the form $A \rightarrow \Gamma_{\mathcal{D}}(A)$.*

Since $\Gamma_{\mathcal{D}}(\{t\}) = \{t' \in \mathcal{T} \mid t' \preceq t\}$ by definition, if $m_t \models A$, then $A \preceq t$, hence $\Gamma_{\mathcal{D}}(A) \preceq \Gamma_{\mathcal{D}}(\{t\})$, and $m_t \models \Gamma_{\mathcal{D}}(A)$. For $\mathcal{R}_0$, the very definition of m_t ensures the claim.

We collect all closure-based implications into the following set:

$$\mathcal{R}'_{\mathcal{D}} = \bigcup_{\mathcal{C}} \{A \rightarrow t \mid \Gamma_{\mathcal{D}}(A) = \mathcal{C},\ t \in \mathcal{C}\}$$

For use in our algorithms below, we also specify a concrete set of rules among those that come from the closure operator. For each closed set of trees $\mathcal{C}$, consider the set of "immediate predecessors", that is, subsets of $\mathcal{C}$ that are closed, but where no other intervening closed set exists between them and $\mathcal{C}$; and, for each of them, say $\mathcal{C}_i$, define:

$$F_i = \{t \mid t \preceq \mathcal{C},\ t \not\preceq \mathcal{C}_i\}$$

Then, we define $\mathcal{H}_{\mathcal{C}}$ as a family of sets of trees that fulfill two properties: each $H \in \mathcal{H}_{\mathcal{C}}$ intersects each F_i, and all the $H \in \mathcal{H}_{\mathcal{C}}$ are minimal (with respect to $\preceq$) under that condition.

We pick now the following set of rules $\mathcal{R}_{\mathcal{D}}$,

$$\mathcal{R}_{\mathcal{D}} = \bigcup_{\mathcal{C}} \{H \rightarrow t \mid H \in \mathcal{H}_{\mathcal{C}},\ t \in \mathcal{C}\}$$

as a subset of the much larger set of rules $\mathcal{R}'_{\mathcal{D}}$ defined above, and state our main result:

Theorem 3.2 *Given the dataset $\mathcal{D}$ of trees, the following propositional formulas are logically equivalent:*

i/ the conjunction of all the Horn formulas satisfied by all the models m_t for $t \in \mathcal{D}$;

ii/ the conjunction of $\mathcal{R}_0$ and all the propositional translations of the formulas in $\mathcal{R}'_{\mathcal{D}}$;

iii/ the conjunction of $\mathcal{R}_0$ and all the propositional translations of the formulas in $\mathcal{R}_\mathcal{D}$.

Proof Note first that i/ is easily seen to imply ii/, because Lemma 3.1 means that all the conjuncts in ii/ also belong to i/. Similarly, ii/ trivially implies iii/ because all the conjuncts in iii/ also belong to ii/. It remains to argue that the formula in iii/ implies that of i/. Pick any Horn formula $H \rightarrow v$ that is satisfied by all the models m_t for $t \in \mathcal{D}$: that is, whenever $m_t \models H$, then $m_t \models v$. Let $v = v_{t'}$: this means that, for all $t \in \mathcal{D}$, if $H \preceq t$ then $t' \preceq t$, or, equivalently, $t' \in \Gamma_\mathcal{D}(H)$. We prove that there is $H' \preceq H$ that minimally intersects all the sets of the form

$$F_i = \{t \mid t \preceq \mathcal{C},\, t \not\preceq \mathcal{C}_i\}$$

for closed $\mathcal{C} = \Gamma_\mathcal{D}$, and for its set of immediate predecessors $\mathcal{C}_i$. Once we have such an H', since $t \in \mathcal{C}$, the rule $H' \rightarrow t$ is in $R_\mathcal{D}$. Together with $\mathcal{R}_0$, their joint propositional translations entail $H \rightarrow t$: an arbitrary model making true H and fulfilling $\mathcal{R}_0$ must make H' true because of $H' \preceq H$ and, if $H' \rightarrow t$ holds for it, t is also true in it. Since $\mathcal{R}_0$ and $H' \rightarrow t$ are available, $H \rightarrow t$ holds.

Therefore, we just need to prove that such $H' \preceq H$ exists. Note that H already intersects all the F_i: $H \preceq \Gamma_\mathcal{D}(H) = \mathcal{C}$; suppose that for some proper predecessor $\mathcal{C}_i$, H does not intersect F_i. This means that $t \preceq \mathcal{C}_i$ for all $t \in H$, and thus, the smallest closed set above H, that is, $\Gamma_\mathcal{D}(H) = \mathcal{C}$, must be below the closed set $\mathcal{C}_i$ or coincide with it, and neither is possible.

Hence, it suffices to consider all the sets of trees H'', where $H'' \preceq H$, that still intersect all the F_i. This is not an empty family since H itself is in it, and it is a finite family; therefore, it has at least one minimal element (with respect to $\preceq$), and any of them can be picked for our H'. This completes the proof.

4 On Finding Implicit Rules for Subtrees

We formally define inplicit rules as follows:

Definition Given three trees t_1, t_2, t_3, we say that $t_1 \wedge t_2 \rightarrow t_3$ is an *implicit Horn rule* (abbreviately, an *implicit rule*) if for every tree t it holds

$$t_1 \preceq t \;\wedge\; t_2 \preceq t \;\leftrightarrow\; t_3 \preceq t.$$

We say that two trees t_1, t_2, have implicit rules if there is some tree t_3 for which $t_1 \wedge t_2 \rightarrow t_3$ is an implicit Horn rule.

A natural generalization having more than two antecedents could be considered; we circunscribe our study to implicit rules of two antecedents.

The aim of the next definitions is to provide formal tools to classify a rule as implicit.

Definition A tree c is a *minimal common supertree* of two trees a and b if $a \preceq c$, $b \preceq c$, and for every $d \prec c$, either $a \not\preceq d$ or $b \not\preceq d$.

In the example of implicit rule given in the introduction, the tree on the right of the implication sign is a minimal common supertree of the trees on the left.

Definition Given two trees a, b, we define $a \oplus b$ as the minimal common supertree of a and b.

As there may be more than one minimal common supertree of two trees, we choose the one with smallest natural representation, as given in (Balcázar et al., 2007b) to avoid the ambiguity of the definition.

Definition A component c_1 of a is *maximum* if any component c_2 of a satisfies $c_2 \preceq c_1$, and it is *maximal* if there is no component c_2 in a such that $c_1 \prec c_2$.

Note that a tree may not have maximum components but, in case it has more than one, all of them must be equal. The following facts on components will be useful later on. The proofs are not difficult and will be provided in a later version of this paper, for the sake of lack of space.

Lemma 4.1 *If a tree has no maximum component, it must have at least two maximal incomparable components.*

Lemma 4.2 *Two trees have implicit rules if and only if they have a unique minimal common supertree.*

Using Lemma 4.2 we can compute implicit rules in an algorithmically expensive way, obtaining minimal common supertrees, which has quadratic cost. To avoid that, we propose several heuristics to speed up the process.

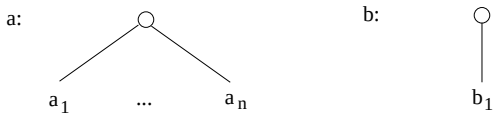
A simple consequence of these lemmas is:

Corollary 4.3 *All trees a, b such that $a \preceq b$ have implicit rules.*

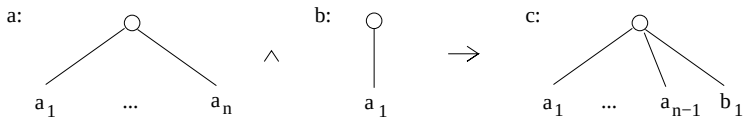
One particularly useful case where we can formally prove implicit rules, and which helps detecting a large amount of them in real-life dataset mining, occurs when one of the trees has a single component.

Theorem 4.4 *Suppose that a and b are two incomparable trees, and b has only one component. Then they have implicit rules if and only if a has a maximum component which is a subtree of the component of b.*

Proof Suppose that a and b are two incomparable trees as described in the statement: a has components $a_1, \ldots, a_n$, and b has only the component b_1. We represent their structures graphically as



Suppose that a has a maximum component which is a subtree of b_1. Without loss of generality, we can assume that a_n is such a component. Then, we claim that $a \wedge b \rightarrow c$ is an implicit rule, where c is a tree with components $a_1, \ldots, a_{n-1}$, and b_1. That is,



To show that this is actually an implicit rule, suppose that, for some tree x, $a \preceq x$ and $b \preceq x$. From the fact that $a \preceq x$, we gain some insight into the structure of x: it must contain components where a's and b's components can map, and so, there must be at least n components in x. So, let $x_1, \ldots, x_m$ be the components of x, with $m \geq n$, and let us suppose that $a_i \preceq x_i$ for every i such that $1 \leq i \leq n$.

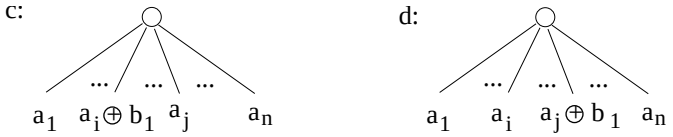
Since b is also a subtree of x, b_1 must be a subtree of some x_i with $1 \leq i \leq m$. We now show that, for every possible value of i, c must be a subtree of x and then, $a \wedge b \rightarrow c$ is an implicit rule:

- If $i \geq n$, then $a_k \preceq x_k$ for all $k \leq n-1$, and $b_1 \preceq x_i$.
- If $i < n$, then
  - $a_k \preceq x_k$ for $k \neq i$ and $1 \leq k \leq n-1$
  - $a_i \preceq a_n \preceq x_n$
  - $b_1 \preceq x_i$

In both cases, $c \preceq x$, and we are done.

To show the other direction, let us suppose that a does not have a maximum component which is a subtree of b_1. We will show that, in this case, there are two different minimal common supertrees of a and b. Then, by Lemma 4.2, we will get the desired conclusion. The previous condition on maximal components can be split into two possibilities:

1. Tree a does not have a maximum component. By Lemma 4.1, there must be two maximal components of a which are incomparable, let us say a_i and a_j. Now we claim that the two trees c and d in the following figure are two different minimal common supertrees of a and b:



In the first place, we show that c and d are different. Suppose they are equal. Then, since b_1 cannot be a subtree of any a_k, $1 \leq k \leq n$ (because a and b are assumed to be incomparable), the components containing b_1 must match. But then, the following multisets (the rest of the components in c and d) must be equal:

$$\{a_l \mid 1 \leq l \leq n \ \wedge \ l \neq i\} = \{a_l \mid 1 \leq l \leq n \ \wedge \ l \neq j\}.$$

But the equality holds if and only if $a_i = a_j$, which is false. Then $c \neq d$.

Second, we show that c contains a and b minimally. Call $c_1, \ldots, c_n$ to the components of c in the same order they are displayed: $c_k = a_k$ for all $k \leq n$ except for $k = i$, for which

$c_k = a_i \oplus b_1$. Suppose now that we delete a leaf from c, getting $c' \prec c$, whose components are $c'_1, \ldots, c'_n$ (which are like the corresponding c_k's except for the one containing the deleted leaf). We will see that c' does not contain a or b by analyzing two possibilities for the location of the deleted leaf, either (a) in the component $c_i = a_i \oplus b_1$ or (b) in any other component:

(a) Suppose that the deleted leaf is from $c_i = a_i \oplus b_1$ (that is, $c'_i \prec c_i$). Then, either $a_i \not\preceq c'_i$ or $b_1 \not\preceq c'_i$. In the case that $b_1 \not\preceq c'_i$, we have that $b \not\preceq c'$ since b_1 is not included in any other component. So, suppose that $a_i \not\preceq c'_i$. In this case, consider the number s of occurrences of a_i in a. Since a_i is a maximal component, the occurrences of a_i in a are the only components that contain a_i as a subtree. Therefore, the number of components of c that contain a_i is exactly s, but it is $s-1$ in c' due to the deleted leaf in $a_i \oplus b_1$. Then, $a \not\preceq c'$.

(b) Suppose now that the deleted leaf is from c_k for $k \neq i$. In this case, it is clear that $a_k \not\preceq c'_k$, but we must make sure that $a \not\preceq c'$ by means of some mapping that matches a_k with a component of c' different from c'_k. For contradiction, suppose there exists such a mapping, that is, for some permutation π from the symmetric group of n elements, we have $a_m \preceq c'_{\pi(m)}$ for every $m \leq n$. Let l be the length of the cycle containing k in the cycle representation of π (so, we have $\pi^l(k) = k$, and has a value different from k for exponents 1 to $l-1$). We have that

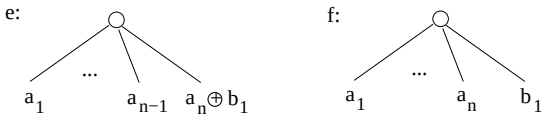
$$a_k \preceq a_{\pi(k)} \preceq a_{\pi^2(k)} \preceq \ldots \preceq a_{\pi^{l-1}(k)}$$

since for every a_m in the previous chain except the last one, if $\pi(m) \neq i$, then $a_m \preceq c'_{\pi(m)} = a_{\pi(m)}$; while if $\pi(m) = i$, then $a_m \preceq a_{\pi(m)}$ because a_i is a maximum component.

From the previous chain of containments, we conclude that $a_k \preceq a_{\pi^{l-1}(k)}$. But $a_{\pi^{l-1}(k)} \preceq c'_{\pi^l(k)} = c'_k$. Putting it together, we get $a_k \preceq c'_k$, which is a contradiction. Therefore, $a \not\preceq c'$.

Now, from (a) and (b), we can conclude that c is a minimal common supertree of a and b. Obviously, the same property can be argued for d in a symmetric way, and since c and d are different, Lemma 4.2 implies that a and b cannot have implicit rules.

2. Tree a has maximum components but they are not subtrees of b_1. We consider now the following trees:



We will show (a) that tree e is a minimal common supertree of a and b, and (b) that tree f is a common supertree of a and b and does not contain e. From (a) and (b), we can conclude that a and b must have two different minimal common subtrees. Take e as one of them. For the other one, let f' be a tree obtained from f by deleting leaves until it is minimal (that is, deleting one more leave would not contain a or b). Since $e \not\preceq f$ (from point (b)), it holds that $e \not\preceq f'$. On the other hand, if we had $f' \preceq e$, since e is minimal, we would have $e = f'$, and then $e \preceq f$,

which contradicts point (b). Therefore, e and f' must be two incomparable minimal common supertrees of a and b, and the theorem follows. To complete the proof, it is only left to show:

(a) *Tree e is a minimal common supertree of a and b.* Note that the proof in previous case 1, showing that c is a minimal common supertree of a and b, applies to e as well. The argument for c was based on the maximality of a_i, but a_n is maximum in e, and then it is also maximal, so the proof applies.

(b) *Tree f is a common supertree of a and b, and does not contain e.* Clearly by definition, f is a common supertree of a and b. Now, we will argue that $e \not\preceq f$. For this inclusion to be true, $a_n \oplus b_1$ should be a subtree of some component of f. It cannot be a subtree of one of the a_k's components ($k \leq n$) since then $b_1 \preceq a_k$ and $b \preceq a$, which is false. On the other hand, $a_n \oplus b_1$ cannot be a subtree of b_1 neither, because that would mean that $a_n \preceq b_1$, which is false in this case. Therefore, f does not contain e.

Since we have proved the existence of two minimal common supertrees also for this case, a new application of Lemma 4.2 completes the proof.

Corollary 4.5 *Two trees with one component each have implicit rules if and only if they are comparable.*

In fact, one fragment of the argumentation of this theorem can be also applied directly as well to some cases that do appear in practice:

Definition Given two trees a, b, we denote by $a + b$ the tree built joining the roots of all components of a and b to a single root node.

Definition Given two trees a and b, tree a with components $a_1, \cdots, a_n$ and tree b with components $b_1, \cdots, b_k$, and $n \geq k$, we denote by $a \uplus b$ the tree built recursively by joining the trees $a_i \uplus b_i$ for $1 \leq i \leq k$, and a_i for $k < i \leq n$, to a single root node. If b has only a node then $a \uplus b = a$. In case that $n < k$, $a \uplus b$ is defined as $b \uplus a$.

Proposition 4.6 *The rule $a \wedge b \rightarrow c$ is not an implicit rule if $c \not\preceq a + b$ or $c \not\preceq a \uplus b$.*

Proof If $c \not\preceq a + b$ or $c \not\preceq a \uplus b$, then $a + b$ or $a \uplus b$ are supertrees of a and b that are not supertrees of c and by the definition of implicit rule, the rule $a \wedge b \rightarrow c$ is not implicit.

Using Proposition 4.6, we have implemented an additional recursive heuristic that can be explained as follows: for every rule $a \wedge b \rightarrow c$ we build $a + b$ and $a \uplus b$ and if we realize that one of them is not a supertree of c, then the rule is not implicit.

5 Experimental Validation

We tested our algorithms on two real datasets. The first one is CSLOGS Dataset (Zaki (2002)). It consists of web logs files collected over one month at the Department of Computer

Science of Rensselaer Polytechnic Institute. The logs touched 13.361 unique web pages and CSLOGS dataset contains 59.691 trees. The average size of the trees is 12.

The second dataset is Gazelle, a dataset from KDD Cup 2000 (Kohavi et al., 2000). This dataset is a web log file of a real internet shopping mall (gazelle.com), has size 1.2GB and contains 216 attributes. We use the attribute 'Session ID' to associate to each user session a unique tree. The trees record the sequence of web pages that have been visited in a user session. Each node tree represents a content, assortment and product path. Trees are not built using the structure of the web site, instead they are built following the user streaming. Each time a user visits a page, if he has not visited it before, we take this page as a new deeper node, otherwise, we backtrack to the node this page corresponds to, if it is the last node visited on a concrete depth. The resulting dataset consists of 225.558 trees.

On these datasets, we have computed association rules following our method. We have then analyzed a number of issues. First, we have checked how many redundant rules could be avoided by some more sophisticated rule production system along the lines of a Duquenne-Guigues basis; however, the structure of these datasets leads to little or no redundancy for this reason, and we omit further discussion of this consideration.

Then, we have implemented an implicit rule detection step based on all the criteria described in the previous section. Timing considerations are rather irrelevant, in that the time overhead imposed by this implicit rule detection step is reasonably low. We compare the number of rules obtained, the number of implicit and not implicit detected rules, and the number of non implicit rules. Figure 1 shows the results for the CSLOGS dataset, and the Gazelle dataset. We observe that when the minimum support of the closed frequent subtrees decreases, the number of rules increases and the number of detected rules decreases. The number of detected rules depends on the dataset and on the minimum support. As an example, our method detects whether a rule is implicit or not in 91% of the rules obtained from CSLOGS dataset with a support of 7.500, and 32% of the rules obtained from Gazelle Dataset with a support of 500. The number of non implicit rules are more than 75% in the two datasets.

6 Conclusions

We have developed, on the basis of a closure operator on sets of trees for a given dataset, studied in our previous work, a new form of implication (or deterministic association rule) among trees. We have presented a mathematical characterization and proposed a method to obtain a basis.

Then we have discussed why the particular combinatorics of our application of the basis still lead to redundant information in the output: implicit rules that are constructed by our method but, actually, due to the combinatorics of the trees, will hold in all datasets and speak nothing about the dataset under analysis.

Whereas a complete characterization of such redundancies is, so far, out of the scope of our work, we have been able to provide an exact characterization for one particular case, where one of the two trees involved in the antecedents has a single component. We have demonstrated, through an implementation and an empiric analysis on real-life datasets, that our development offers a good balance between mathematical sophistication and efficiency in the detection of implicit rules, since with just our characterization and two heuristics we catch a large ratio of implicit rules.

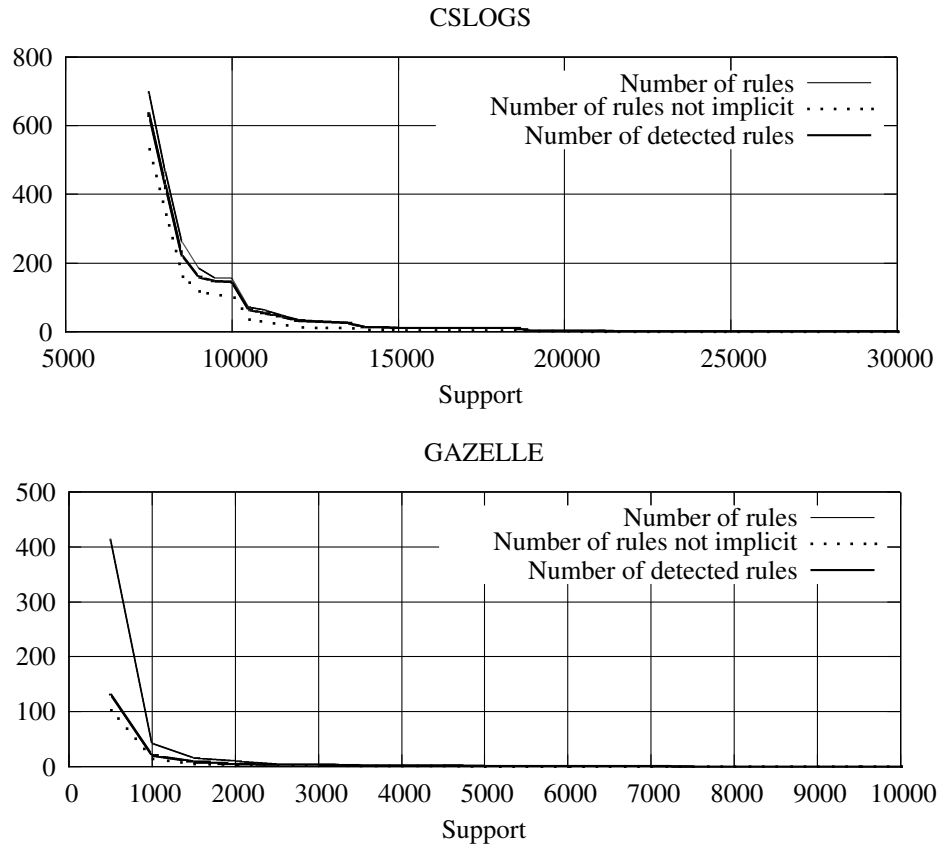


FIG. 1 – *Real data experimental results on CSLOGS and Gazelle datasets*

References

Balcázar, J. L., A. Bifet, and A. Lozano (2006). Intersection algorithms and a closure operator on unordered trees. In *MLG 2006, 4th International Workshop on Mining and Learning with Graphs*.

Balcázar, J. L., A. Bifet, and A. Lozano (2007a). Mining frequent closed unordered trees through natural representations. *Proceedings of the 15th International Conference on Conceptual Structures (ICCS 2007)*.

Balcázar, J. L., A. Bifet, and A. Lozano (2007b). Subtree testing and closed tree mining through natural representations. *Workshop Advances in Conceptual Knowledge Engineering*.

Balcázar, J. L. and G. C. Garriga (2007a). Characterizing implications of injective partial orders. In *Proceedings of the 15th International Conference on Conceptual Structures (ICCS*

2007).

Balcázar, J. L. and G. C. Garriga (2007b). Horn axiomatizations for sequential data. *Theoretical Computer Science 371*(3), 247–264.

Chi, Y., R. Muntz, S. Nijssen, and J. Kok (2005a). Frequent subtree mining – an overview. *Fundam. Inf. 66*(1-2), 161–198.

Chi, Y., Y. Xia, Y. Yang, and R. Muntz (2005b). Mining closed and maximal frequent subtrees from databases of labeled rooted trees. *IEEE Transactions on Knowledge and Data Engineering 17*(2), 190–202.

Kohavi, R., C. Brodley, B. Frasca, L. Mason, and Z. Zheng (2000). KDD-Cup 2000 organizers' report: Peeling the onion. *SIGKDD Explorations 2*(2), 86–98.

Pfaltz, J. L. and C. M. Taylor (2002). Scientific knowledge discovery through iterative transformations of concept lattices. In *Workshop on Discrete Math. and Data Mining at SIAM DM Conference*, pp. 65–74.

Termier, A., M.-C. Rousset, and M. Sebag (2004). DRYADE: a new approach for discovering closed frequent trees in heterogeneous tree databases. In *ICDM*, pp. 543–546.

Wild, M. (September 1994). A theory of finite closure spaces based on implications. *Advances in Mathematics 108*, 118–139(22).

Zaki, M. J. (2002). Efficiently mining frequent trees in a forest. In *8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Résumé

Nous proposons une manière à extraire des règles d'association de haute confiance d'ensembles de données se composant d'arbres non étiquetés. Les antécédents sont obtenus par un calcul apparenté à un transversal d'hypergraphe, tandis que les conséquents se suivent d'une application des opérateurs de fermeture sur les arbres non étiquetés développés dans des travaux précédents des auteurs. Nous discutons en plus détail le cas des règles trivialement valides, puisque celles-ci sont plus complexes que dans le cas des itemsets, étant donné que nous ne travaillons plus avec un trellis.