International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 8 (2016) pp. 052-056
© MIR Labs, www.mirlabs.net/ijcisim/index.html

Comparison between two declarative approaches to solve the problem of Pattern Mining in Sequences

Akram Rajeb [1], Zied Loukil [1] and Abdelmajid Ben Hamadou [1]

[1] University of Sfax, MIRACL Laboratory
Sfax, Tunisia

akram.isimg@gmail.com
{abdelmajid.benhamadou; zied.loukil}@gmail.com

***Abstract*: Declarative approaches to solving the problem of Pattern Mining in Sequences (EMS) constitute an interesting technique transforming the EMS problem into another equivalent NP-Complete problem and trying to solve it by using specialized solvers.**

In this paper, we will present two existing declarative approaches the first transforms EMS problems into SAT problems (Boolean Satisfiability problems) and the second into CSP (Constraint Satisfaction Problems). We will focus on the frequent and closed pattern mining in sequences (F-CPS) and we will compare the obtained results for these problems.

Keywords: SAT, CSP, constraint programming, pattern mining, declarative approaches

I. Introduction

Pattern Mining in Sequences (EMS) is an important topic in data mining including many problems in industry, marketing (cross-selling and up-selling), and biology (DNA sequences analysis)... It aims to extract patterns that frequently appear in the targeted sequence.
In this paper, we will focalize in enumerating patterns that appear at least λ times in sequence. It is an NP-complete problem [1] and there are many techniques and approaches trying to solve it as rapidly as possible.

Declarative approaches constitute a set of methods trying to solve this problem by transforming it into other NP-complete problems such as SAT problems (Boolean Satisfiability problems) [2] or CSP (constraint satisfaction problems) [3, 4, 5]. This transformation aims to reduce of the number of patterns to be analyzed, by focusing on the needs, and the elimination of patterns that do not satisfy the needs.
This transformation is possible in polynomial time, thanks to the NP-completeness propriety of all these problems and it allows taking profit of the performance of the other solvers with different kinds of heuristics and filtering methods to solve EMS problems.

In this paper, we will compare the results of several works transforming pattern mining problems into SAT and CSP problems and trying to resolve them by the associated solvers. We will focus on the frequent and closed pattern mining in sequences (F-CPS).

This paper is organized as follows. After this introduction, the second section provides an introduction to the main principles of frequent and closed pattern mining problems in sequences, constraint satisfaction problems (CSP) and Boolean Satisfiability problems (SAT). The third section presents a declarative approach to solve the problems of frequent and closed pattern mining in sequences by encoding them into SAT problems or into CSPs. we will compare results of the two approaches. Finally, experimental results are discussed before concluding.

II. Preliminaries

A. *Main definitions in pattern mining problems*

1) *Sequence of items*

Let Σ be an alphabet built on a finite set of symbols. A sequence S is a succession of characters $S_1 \dots S_n$ such that $Si \in \Sigma$ $\{1 \dots m\}$ represents the character at position i in S. The length of the string S is denoted by $| S | = n$. We denote $\theta = \{1 \dots m\}$ as the set of positions of characters in S. A wildcard is an additional character noted o not belonging to Σ ($o \notin \Sigma$) that can match any symbols of the alphabet.

2) *Pattern*

A pattern over Σ is a string $p = p_1 \dots p_m$ where $p_1 \in \sum$, $p_m \in \sum$ and $p_i \in \sum \cup \{o\}$ for i =2…..m-1 (Started and ends with a solid character).

3) *Occurrence ($\mathcal{L}$)*

We consider the location list $\mathcal{L}_z \subseteq \{1 \dots n\}$ as the set of all the position on s at which z occurs.

4) *Frequent pattern*

Let S be a sequence and a pattern p, λ is a positive number called quorum and p is a frequent pattern in S when $|L_S(p)| \geq \lambda$ .The set of all patterns of S for the quorum λ is denoted: M_S^λ.

5) *Closed pattern*

In a sequence of items, a frequent pattern p is considered closed if for any frequent pattern q satisfying $q \supset p$, there is no integer d such $\mathcal{L}_s \subseteq_s (q) = \mathcal{L}_s \subseteq_s (p) + d$,
Where
$\mathcal{L}_s \subseteq_s (p) + d = \{l + d \mid l \in \mathcal{L}_s \subseteq_s (p)\}$.

6) *Example*

S= AABCAAB
λ = 2
The set of frequent patterns:

{P1=A, P2=B, P3=A000A, P4=A0B, P5=AA, P6=AAB and P7=AB}

$\mathcal{L}_s$ (P1) = {1, 2, 5, 6}
$\mathcal{L}_s$ (P2) = {3, 7}
$\mathcal{L}_s$ (P3) = {1, 2}
$\mathcal{L}_s$ (P4) = {1, 4}
$\mathcal{L}_s$ (P5) = {1, 4}
$\mathcal{L}_s$ (P6) = {1, 4}
$\mathcal{L}_s$ (P7) = {2, 4}

P1 ⊂ P3, P1 ⊂ P4, P1 ⊂ P4, P1 ⊂ P5, P1 ⊂ P6 and P1 ⊂ P5, but the support of P1 is bigger in size, so even shifting can’t fall on the support of P1.
So, the P1 is a frequent closed pattern.
In the same way, P3 and P4 are considered closed frequent patterns.

B. *Boolean Satisfiability (SAT) problems*

1) *Literal*

A literal is a positive or negative propositional variable.

2) *Clause*

A clause is a set of disjunction of literals.

3) *Conjunctive normal form*

A conjunctive normal form (CNF) is a set of the conjunction of clauses.
In any propositional formula, there is an equivalent CNF, and we call it a SAT instance.

4) *Interpretation and model*

The interpretation of a propositional formula is an assignment of all its variables to true or false.
A model is an interpretation which makes the global formula true.
In SAT instance, a model is an assignment of its literals to true or false which makes all its clauses true.

5) *The SAT problem*

The SAT problem consists in determining if a SAT instance admits a model or not.
It is an NP-complete problem [6] and there are many algorithms to solve SAT instances as quick as possible. These algorithms are implemented into different solvers and the most of them can read SAT instances encoded in DIMACS format.

C. *Constraint satisfaction problems (CSP)*

A CSP is specified by a set of variables, a domain, which maps by variable to a set of values and a set of constraint.
Formally, a CSP is defined by a triple (X, D, C), where:
$X = \{X_1, X_2 \ldots X_n\}$ is the set of variables of the problem
D is the set of the domains of each variable
C is a set of constraints $\{C_1, C_2 \ldots C_n\}$. Each constraint $C_i \in C$ is a constraint over some subset of variables.

1) *Arity of the CSP*

The size of this subset is known as the arity of the constraint.
Non-binary CSPs are CSPs that contain constraints with arity greater than 2.

2) *Solution of CSP*

Solving a CSP consists of searching an assignment of values to all its variables in their domains satisfying all the constraints. It is considered an NP-complete problem when all the domains are finite and all the constraints are with arity greater than 1.

3) *CSP solvers*

Algorithms for solving CSPs are called solvers. Some of these solvers have been integrated a libraries into programming languages, thus defining a new programming paradigm called constraint programming: to solve a CSP with constraint programming language, it is sufficient to specify constraints; their resolution is supported automatically (without needing a program) by constraint solvers integrated language.

It is important to correctly describe the problem with the respect of all the rules defined by the solver and its programming language before trying to solve it. That makes the comparison between CSP solvers quite difficult.
For this reason, several formalisms are defined to describe constraint-based problems such as MinZinc [7], XCSP [8, 9]… and the majority of CSP solvers are able to read problems defined with their formalisms.

III. Frequent and closed patterns mining in a sequence (F-CPS)

In this section, we introduce the problem of frequent and closed patterns mining in a sequence. Let us first give some preliminary definitions and notations.

A. A comparison between SAT and non-binary CSP

A SAT problem can be encoded (in polynomial time) as a binary or non-binary CSP because both of the two problems are NP-complete. For this reason, we will only describe the transformation of EMS into a CSP model presented in [3, 4, 5].

1) Variables

We introduce two types of variables:
P = {p_1, p_2… p_m} represents the candidate pattern.
B = {b_1, b_2… b_n} represents the support (p), it's an integer variable: b_k =0 if the pattern is not located in S at the position k, 1 otherwise.

2) Domains

Dom (p_i) = ∑ U {o}
Dom (b_k) = {0, 1}

3) Constraints

$$\mathbf{p_i} \neq \mathbf{o} \qquad \text{(Ct1)}$$

For all $1 \le k \le n$
For all $1 \le i \le m$

$$\mathbf{p_i \neq o \rightarrow (p_i \neq S_{i+k-1} \rightarrow b_k)} \qquad \text{(Ct2)}$$

$$\sum_{k=1}^{n} \mathbf{b_k} \le \mathbf{n - \lambda} \qquad \text{(Ct3)}$$

We run experiments on PCs with Intel i7 processors and 6GB of RAM. In [3, 4, 5], we encoded the obtained model into XCSP format so it is possible to use all the compatible solvers using this formalism. We used the solver CHOCO: a library that implements the basic tools for the constraint programming: domain management, constraint propagation, global process and local search. This library has been implemented in the project OCRE for the purpose is to offer a constraint tool for Research and education (OCRE). It is built in a propagation mechanism based on events with backtrack structures.

In our experiments, we used two data sets with different size, the first one is DS_sz-50[1] (size of sequence = 50) and the second is DS_sz_100 (size of sequence =100).
The problem of enumerating all frequent patterns is expressed by the constraints Ct1, Ct2 and Ct3.

We commence by running our program with the 3 above constraints (ct1, ct2 and ct2) for finding only the frequent patterns.

In the first experiment, we compare the performance of our approach CSP against the SAT approach proposed in [2] used the first dataset (DS_sz_50[1]).

In the second experiment, we compare our approach CSP against the SAT approach proposed in [3, 4, 5] used the second dataset (DS_sz_100[1])

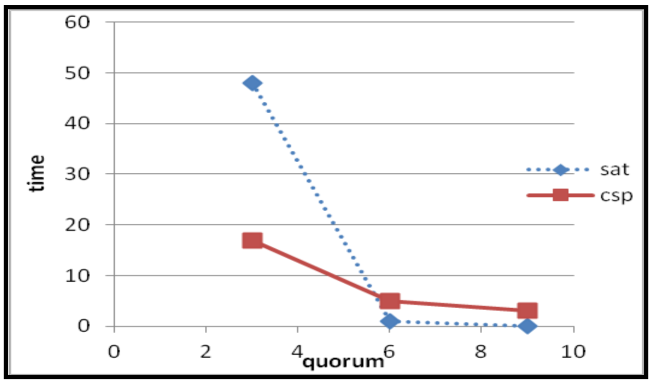

Figure 1.SAT Vs CSP on DS_sz_50 sequence

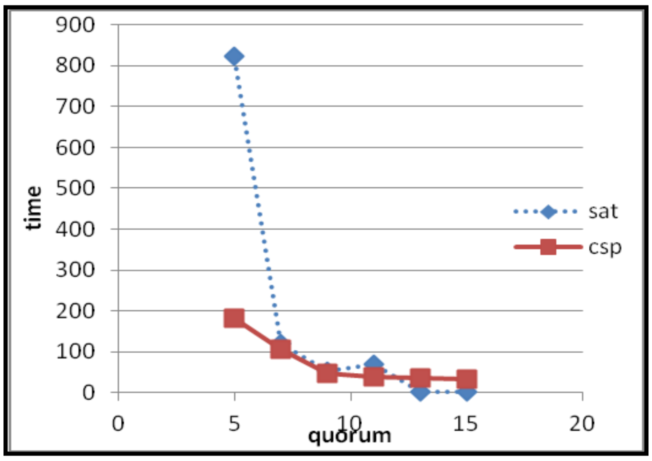

Figure 2. SAT Vs CSP on DS_sz_100 sequence

In the above experiment we have shown that our approach more efficient than SAT approach proposed in [2].

We have also seen, when the quorum decreases, and the size of the sequence increases the CPU time in the SAT approach increases very quickly compared with our approach, the CPU time increases slightly.

Let us now introduce the closeness constraint:

$$\bigwedge_{k=1}^{n} (\,]\,(\,]\,\boldsymbol{b_k} \vee \boldsymbol{S_{k+i-1}} = \boldsymbol{a}) \bigvee \boldsymbol{p_i} = \boldsymbol{a}$$
$$\forall\, \mathbf{1} \le \boldsymbol{i} \le \boldsymbol{m}\,, \boldsymbol{a} \in \sum \qquad \text{(Ct4)}$$

[1]**http://www.biomedcentral.com/1471-2105/11/175/additional/**

$$\neg\left(\bigwedge_{i=m-j}^{m} p_i = o\right) \bigvee \neg\left(\bigwedge_{k=1}^{n}(\neg b_k \vee S_{k-j-1} = a)\right)$$

$$for\ 0 \leq j \leq m-2\ , a \in \sum \qquad \text{(Ct5)}$$

```
Algorithm: Closed frequent patterns mining
Inputs: Sequence, Quorum: λ
Output: {} closed frequent patterns: P
Begin
    1- Solve the model M
    2- While there are solutions
        View solution
        Skip to next solution
      End while
End.
```

We present now the next experiments by introducing the closure constraint.

Quorum	frequent closed patterns		frequent patterns	
	Pattern number			
	DS_sz_50 sequence	DS_sz_100 sequence	DS_sz_50 sequence	DS_sz_100 sequence
2	34	24	100	80
4	6	6	12	21
6	2	3	2	2

Table1. Comparison between frequent patterns and the frequent closed patterns

To sum up, in the above experiments, we have shown that the new encoding that uses the closeness constraint is more interesting than the first. Using this constraint, the number of patterns is reduced. As an example, where λ= 2, there are 100 patterns which generated for the case of frequent patterns, but for the new one, (i.e. closed frequent patterns) the number of patterns is reduced to 34.

We now compare the performance of the CSP for closed-frequent pattern extraction with SAT proposed in [2].

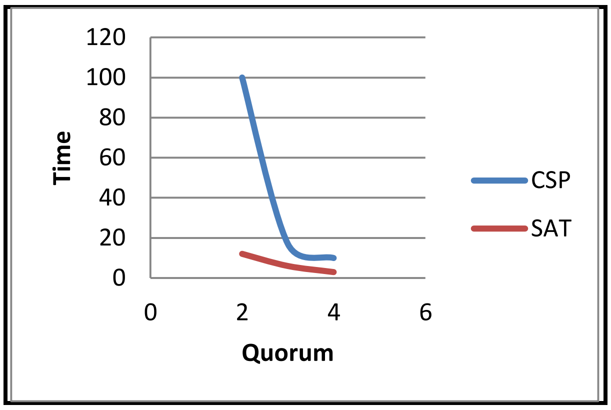

Figure3. SAT-closed Vs CSP-closed on DS_sz_50 sequence

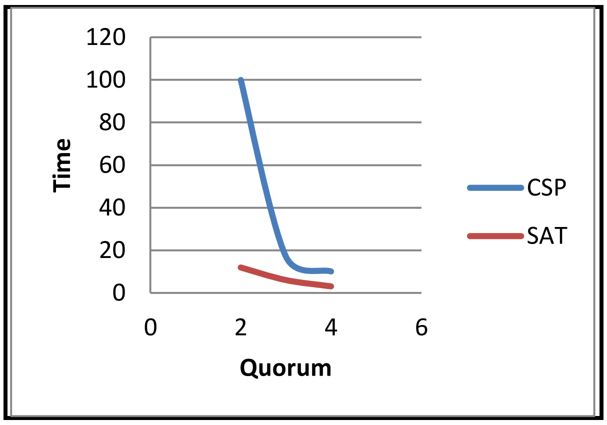

Figure4. SAT-closed Vs CSP-closed on DS_sz_100 sequence

In the above experiment we have shown the SAT is better than CSP when introduced closed constraints.

The introduction of closed constraint in this CSP encoding decreases the performance of our CSP. The latter results can be explained by the large arity of the last constraint. Take for example when the size of the sequence is 100 and the quorum is 2 so the constraint closing arity becomes 298.

The last experimental study is in accordance with the theoretical study presented in [10], which proved that the performance of the non-binary CSP encoding depends on the arity of constraint.

IV. Conclusion

The main idea that constraint programming can offer to data mining is its declarative approach. The use of a high-level declarative language allows us to formulate many problems in the same framework. This is uncommon in data mining, where procedural approaches are the norm.

Constraint programming achieves a greater flexibility because every constraint can independently reduce the search tree.

The disadvantages of the SAT approach are the problem of size: the number of constraints and the number of variables because of the spread of variables.

The CSP approach can solve this problem, but in the non-binary transformation SAT to CSP encoding the performance depends on the arity of constraint.

In this paper, we proposed an experimental study that compares the SAT-Based approach for enumerating frequent, closed patterns in sequences with our CSP-Based approach. Besides, our encoding confirms the theoretical study proved in [11].

References

[1] E Boros, V Gurvich, L Khachiyan, K Makino : “On the complexity of generating maximal frequent and minimal infrequent sets”. in STACS 2002, Springer, 2002.

[2] E.Coquery, S.Jabbour, L.Sais and Y.Salhi. “A SAT-Based Approch for Discovering Frequent, Closed and Maximal Pattern in a Sequence”. In proc of the 20th European Conference on Artificial Intelligence ECAI, pp 258-263, 2012.

[3] A. Rajeb, A. Ben Hamdou and Z. Loukil. “On the enumeration of frequent patterns in sequences”. In proc of the Inernational Conference on Artificial Intelligence and Pattern Recognition (AIPR’2014), pp 40–344, 2014.

[4] A. Rajeb, A. Ben Hamdou and Z. Loukil. “A CSP for the Enumeration of Frequent-closed Patterns in Sequences”. In Proc of the International Conference on Artificial Intelligence and Pattern Recognition (AIPR’2015), pp 59-63, 2015.

[5] A. RAJEB, Z. LOUKIL and A. BEN HAMADOU: “Comparison between SAT-Based and CSP-Based approaches to resolve pattern mining problems “.In proc of the International Conference on Hybrid Intelligent Systems (HIS 2015), 2015.

[6] S. Cook, “The complexity of theorem proving procedures” in the Proc of the third annual ACM symposium on Theory of computing, pp 151–158, 1971

[7] N. Nethercote, P. J. Stuckey, R. Becket, S. Brand, G. J..Duck, and G. Tack. “MiniZinc: Towards a Standard CP Modeling Language”. In CP Proceeding, pages 529–543, 2007.

[8] O. Roussel and C. Lecoutre. « XML Representation of Constraint Networks: Format XCSP 2.1”. in CoRR abs/0902.2362. 2009

[9] A. Malapert and C. Lecoutre. « A propos de la bibliotheque de modeles XCSP. » in the proc. of 10èmes Journées Francophones de Programmation par Contraintes (JFPC'14).pp 337-340. 2014.

[10] H. Bennaceur. “The satisfiability problem regarded as a constraint satisfaction”. In Proc. of the 12h European Conference on Artificial Intelligence (ECAI-96), ECAI, pages 155-159, 1996.

[11] J.de Kleer. “A comparison of ATMS and CSP techniques”. In Proc. of the 11h IJCAI. International Joint Conference on Artificial Intelligence, 198.