Towards Interesting Patterns of Hard CSPs with Functional Constraints

Chendong Li
University of Connecticut
371 Fairfield Road
Storrs, CT 06269, USA
chendong.li@engr.uconn.edu

Abstract

The hardness of finite domain Constraint Satisfaction Problems (CSPs) is an important research topic in Constraint Programming (CP) community. In this paper, we study the association rule mining techniques together with rule deduction and propose a cascaded approach to extract interesting patterns of hard CSPs with functional constraints. Specifically, we generate random CSPs, collect controlling parameters and hardness characteristics by solving all the CSP instances. Afterwards, we apply association rule mining with rule deduction on the collected data set and further extract interesting patterns of the hardness of the randomly generated CSPs. As far as we know, this problem is investigated with data mining techniques for the first time.

Introduction

On the one hand, association rule mining (Agrawal et al. 1993) is a very widely employed data mining technique and it has been successfully applied to market basket analysis to determine customer buying patterns, intrusion detection, financial profiles, bioinformatics, web-based applications, and so forth. On the other hand, how to randomly generate hard CSPs (Mitchell 1998) has been faced by the CP community because hard CSPs are highly desired as they can be used to evaluate and benchmark the novel algorithms of solving CSPs.

Recognized as a common constraints class, functional constraints (Van Hentenryck et al. 1992) attract many researchers' great interest (Liu 1995; Zhang et al. 2008) because of their neat properties, especially the bi-functional constraints (Van Hentenryck et al. 1992; David 1993), which are considered as the tractable constraint class. Since functional constraints are usually easier to solve than general constraints due to their particular structure and latent phase transition (Kwan et al. 1998), the task of generating hard CSPs with functional constraints becomes an extremely challenging problem. To conquer this climate, association rule mining technique provides a potentially effective way to investigate hard CSPs and their interesting patterns.

Previous researches on the hardness of CSPs such as (Cheesman et al. 1991; Hogg and Williams 1994; Prcovic 2005; Ansótegui et al.2008) mainly focus on the general CSPs, where no constraint types are specified. Empirical evaluations on the hard CSPs are presented in (Kwan 1996; Frost et al. 1997).

Outline of Proposed Approach

To extract the interesting patterns of hard CSPs with functional constraints, we propose a cascaded approach. In our approach, we first formulate the problem as the association rule mining problem. We apply Apriori algorithm (Agrawal et al. 1993; Agrawal et al. 1994) to discover the interesting association rules. In the following, we employ rule deduction to further extract the patterns of hard CSPs.

Apriori algorithm. Apriori algorithm generates candidate itemsets of length l from itemsets of length l - 1 and then prunes the candidates which have an infrequent sub pattern. It utilizes a tree structure to count frequent itemsets and uses downward closure to prune unnecessary branches. This algorithm terminates if no further augmentation can be made. Two parameters are involved in the Apriori algorithm: *minimum support* used for generating frequent itemsets and the other is *minimum confidence* used for rule derivation.

Rule deduction. Viewed from the perspective of the deduction methods in propositional logic, a rule $X \Rightarrow Y$ is semantically entailed from a set of rules if every dataset where all the rules hold must also satisfy $X \Rightarrow Y$ (Balcázar 2008). Syntactically, the rule $X \Rightarrow Y$ holds if and only if $X \Rightarrow Y$ is derivable from all the rules by applying the following three main Armstrong's Axioms (Ullman and Widom 2001):

Reflexivity. $X \Rightarrow X$

Transitivity. If $X \Rightarrow Y$ and $Y \Rightarrow Z$ then $X \Rightarrow Z$

Augmentation. If $X \Rightarrow Y$ and $Z \Rightarrow W$ then $X, Z \Rightarrow Y, W$

This mechanism is called rule deduction in our approach. Although association rules are probabilistic implications, here we interpret them under the threshold of *confidence* and *support*. Based on the rule deduction, we can obtain much more meaningful rules. Since applying the reflexivity axiom on any rules cannot provide any new information, we mainly apply transitivity axiom and augmentation axiom. Different from redundant rule reduction (Ashrafi et al. 2005), rule deduction can produce new useful rules. With rule deduction techniques, the task of finding the interesting rules from a set of association rules becomes computing the minimum closure of the set of rules.

Experimental Results

In the experiments, we first generate CSP instances randomly based on the combinations of the controlling parameters, which are number of variables *n*, domain size of each variable *d*, number of constraints *e*, number of functional constraints *nf*, tightness of constraints *t* and seeds *seed*. To generate all possible combinations, we employ additional 4 parameters (Specifically, step size of constraints *stpc*, step size of functional constraints *stpfc*, step size of tightness *stpt* and step size of seeds *stps*) to control the size of increasing strides of the corresponding controlling parameters in the batch files.

Given the huge exponential combinations of parameters, even though we have fixed *n* and *d* to 50, we use 2-step-scanning strategy to deal with the huge combinations: 1) scan sparsely to obtain typical distributions of hard CSPs; 2) concentrate on the relatively small distribution space established at step 1). In the experiment, based on computational results of scanning at step 1), we can eliminate the CSPs with e = 100 and e = 222. Furthermore, we can eliminate the CSPs with the tightness intervals of [0.1, 0.30) and (0.75, 0.90]. The second step scanning focuses on the smaller intervals, i.e., the interval of constraints varying from 344 to 710 still with stride size 122; the interval of tightness varying from 0.30 to 0.75 with stride size 0.01.

To verify the hardness of the randomly generated CSPs, we need solve all the CSP instances. In our experiment, we use a general CSP solver, implemented in `C++` based on *Arc Consistency* algorithm (AC 2001/3.1), to solve all the CSPs to collect their significant characteristics, i.e., controlling parameters and the solving time, of the randomly generated problems. The solver is running on a `DELL PowerEdge 1850` (two 3.6 GHz Intel Xeon CPUs) in `Linux`. Based on the running time of the CSPs, we classify the problem instances into three different classes as shown in Table 1.

Table 1: Hardness classifications based on the solving time for each CSP

Case #	Solving Time(s)	Classification
Case 1	(0.00, 5.00)	Easy
Case 2	[5.00, 60.00]	Medium
Case 3	(60.00, $+\infty$)	Hard

Applying the criteria presented in Table 1, we have 266 instances classified as “Medium” or “Hard” from the total 10334 instances. After obtaining solving time and classifications, we use WEKA to (a collection of data mining algorithms implemented in Java, available at `http://www.cs.waikato.ac.nz/ml/weka/`) discretize the data and carry out the association rule mining, which is mainly based on the Apriori algorithm. Typically, we set `bin` value as 20 to discretize the data and the value of *minimum support* is set to 0.1. In our experiments, the selected measure is *lift* and its value is set to 1.1. Under these specifications, the totally generated interesting rules are 62. The typical one is `e=(691.7-710], t=(0.6075-0.628]` $\Rightarrow$ `Classification=Hard`, which indicates if the total constraints of a CSP is more than 692 and their tightness $\in$ (0.6075-0.628], then this CSP is highly likely to be hard.

By applying rule deduction to all the rules, we eventually obtain the interesting patterns of hard CSPs. The patterns reveal that tightness $\in$ (0.6075, 0.628], constraints $\in$ (691.7-710] and functional constraints $\in$ (27.95-30.4](if any), can generate hard CSP instances. These CSP hardness patterns can provide a very useful recommendation for generating hard CSPs. Based on the patterns, one simply set n = 50, d = 50, e = 700, tightness = 0.618, functional constraints = 29 (if any), the hard CSPs (with functional constraints) can be generated. Also slight change on the tightness can yield more interesting CSP instances.

Acknowledgments. The author would like to thank Dr. Yuanlin Zhang for his useful discussion on functional constraints.The complete work can be accessed at `www.engr.uconn.edu/~chl09007/hardcsp.pdf`.

References

Agrawal, R., Imielinski, T., Swami, A. N. 1993. Mining Association Rules between Sets of Items in Large Databases. In *Proc. of SIGMOD,* 207-216.

Agrawal, R., Srikant, R. 1994. Fast Algorithms for Mining Association Rules in Large Databases. In *Proc. of VLDB,* 487-499.

Anstegui, C., Bjar, R., Fernndez, C., Mateu, C. 2008. Generating Hard SAT/CSP Instances Using Expander Graphs. In *Proc. of AAAI,* 1442-1443.

Balczar, J. L. 2008. Deduction Schemes for Association Rules. In *Proc. of ICDS,* LNCS 5255, 124-135.

Cheesman, P., Kanefsky, B., and Taylor, W.M. 1991. Where the Really Hard Problems Are. In *Proc. of IJCAI,* 331-340.

David, P. 1993. When Functional and Bijective Constraints Make a CSP Polynomial. In *Proc. of IJCAI,* vol. 1, 224-231.

Hogg, T., Williams, C. P. 1994. The Hardest Constraint Satisfaction Problems: A Double Phase Transition. *Artif. Intell.* 69:359-377.

Kwan, A. C. M. 1996. Validity of Normality Assumption in CSP Research. In *Proc. of PRICAI,* 253-263.

Kwan, A. C. M., Tsang, E. P. K., Borrett, J. E. 1998. Predicting Phase Transitions of Binary Constraint Satisfaction Problems with Constraint Graph Information. *Intell. Data Anal.* 2(1-4): 45-62.

Liu, B. 1995. Increasing Functional Constraints Need to Be Checked Only Once. In *Proc. of IJCAI,* 119-125.

Mitchell, D. 1998. Hard Problems for CSP Algorithms. In *Proc. of AAAI/IAAI,* 398-405.

Mitchell, D. 1994. Respecting Your Date (I). In *AAAI-94 Workshop on Experimental Evaluation of Reasoning and Search Methods,* 28-31.

Prcovic, N. 2005. Extremal CSPs. In *Proc. of CP,* LNCS 3709, 807-811.

Ullman, J., Widom, J. 2001. *A First Course in Database Systems.* 2 Edition, Prentice Hall.

Van Hentenryck, P., Deville, Y., and Teng, C. M. 1992. A Generic Arc-consistency Algorithm and Its Specializations. *Artif. Intell.* 58:291-321.

Zhang, Y., Yap, R. H.C., Li, C. and Marisetti, S. 2008. Efficient Algorithms for Functional Constraints. In *Proc. of ICLP* LNCS 5366, 606-620.