Phase Transitions in Knowledge Compilation: an Experimental Study*

Jian Gao[1], Minghao Yin[1], and Ke Xu[2]

[1] College of Computer Science
Northeast Normal University, Changchun, 130024, China.
jiangao.cn@hotmail.com; ymh@nenu.edu.cn
[2] State Key Lab. of Software Development Environment
Beihang University, Beijing, 100191, China.
kexu@nlsde.buaa.edu.cn

Introduction and Background

Phase transitions, as a kind of well-known phenomena in artificial intelligence, have attracted a great amount of attention in recent years [1,2]. Many NP-complete problems, such as random SAT and random Constraint Satisfaction Problems (CSPs), have a critical point that separates overconstrained and underconstrained regions, and soluble-to-insoluble phase transition occurs at this critical point, which is always accompanied with the transitions of CPU runtimes. Both systematic search algorithms and local search algorithms suffer an easy-hard-easy pattern when solving those problems. In fact, the easy-hard-easy patterns are not only expressed in terms of the time, but also in terms of the space. That is phase transition in knowledge compilation.

Knowledge compilation [3] is used to compile solutions of a problem into a tractable language. Many target languages of knowledge compilation have been proposed for compiling SAT instances and CSPs. Easy-hard-easy patterns in those languages have been shown in the early studies [4,5]. Schrag and Crawford [4] studied phase transitions in compiling 3-SAT instances to prime implicates (PIs) and showed the critical point occurs when the ratio (r) of #clauses (m) to #variables (n) is around 2.0. While recent studies have proposed many more succinct languages [3], such as Ordered Binary Decision Diagram (OBDD) [6], deterministic, Decomposable Negation Normal Form (d-DNNF) [7] and Deterministic Finite-state Automaton (DFA). Differ from PIs, these languages covert solutions into more compact forms using the property of solution symmetry. In this paper, we investigate easy-hard-easy patterns in empirical results of compiling random SAT and CSP into OBDD, d-DNNF and DFA.

Main Results

First, we show experimental results concerning random 3-SAT instances. Fig. 1 depicts the easy-hard-easy pattern when compiling instances with n=30 into the three target languages, where the number of nodes in the compilation results are used to measure the sizes. The peak points of those curves are with same value of the ratio r,

* A full version of this paper is available at http://arxiv.org/abs/1104.0843. This work is supported by the National Science Foudation of China under grant No. 60803102 and 60973033. Correspondence to: Minghao Yin (Northeast Normal University) or Ke Xu (Beihang University)

which is 1.8. Additionally, random k-SAT instances are also considered. We can obverse that those target languages share the same critical point of the easy-hard-easy pattern, where $r\approx2.6$ for k=4 and $r\approx4.5$ for k=5. Based on those results, we conjecture that all target languages belonging to subsets of Decomposable Negation Normal Form (DNNF) suffer the compilation phase transition with the same critical point. An explanation of this phenomenon is the inherent changes on interchangeable structure of solutions. We observe that the number of interchangeable solutions with respect to 2 variables has a great impact on the sizes of compilation results.

Next, we show the sizes of compilation results increase exponentially as n grows linearly. We convert 3-SAT instances into d-DNNFs, and take 6 values of r uniformly. For each r, we vary n from 10 to 60 at increments of 5. Fig. 2 shows the results with the logarithmic vertical axis. Curves are all nearly linear, so the size grows exponentially in the general cases. As r is close to 1.8, slopes of lines grow larger, and the sizes around phase transition regions grow fastest. Besides, we surmise there also exists a phase transition separates polynomial and exponential sizes. For 3-SAT, the critical point of the polynomial-to-exponential phase transition is around r=0.3.

Furthermore, we show that the easy-hard-easy pattern also exists in compiling random CSPs. We employ RB model [7] to generate random CSP instances. The RB model is described by constraint arity k, the number of variables n, domain size $d=n^{\alpha}$, constraint number $m=rn\ln n$, and the constraint tightness p. We fix k, α, p, and compile CSPs into DFAs. The peak point of DFA sizes is fixed as the number of variables increases. For instance, when k=2, α=1.2, p=0.5, the peak point occurs at r=0.52.

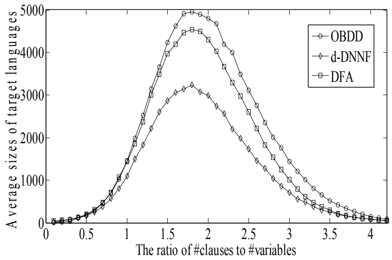

Fig. 1. The easy hard easy pattern

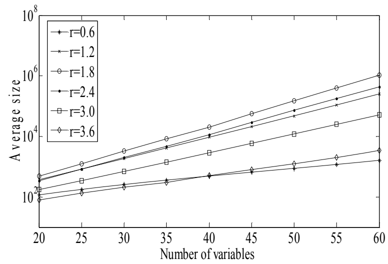

Fig. 2. Exponential increments of sizes

References

1. Cheeseman P., Kanefsky B., Taylor W. M.: Where the Really Hard Problems Are. In: proc. IJCAI'91, pp. 331--340. Morgan Kaufmann Publishers, Inc. (1991)
2. Achlioptas D., Naor A., Peres Y.: Rigorous location of phase transitions in hard optimization problems. Nature 435, 759--764 (2005)
3. Darwiche A., Marquis P.: A Knowledge Compilation Map. J. Artif. Intell. Res. 17, 229--264 (2002)
4. Schrag R., Crawford J. M.: Implicates and prime implicates in Random 3-SAT. Artificial Intelligence, 81, 199--222 (1996)
5. Darwiche A.: A Compiler for Deterministic, Decomposable Negation Normal Form. In: proc. AAAI/IAAI'02: pp. 627--634. AAAI Press (2002)
6. Narodytska N., Walsh T.: Constraint and Variable Ordering Heuristics for Compiling Configuration Problems. In: proc. IJCAI'07, pp. 149--154. Morgan Kaufmann Publishers, Inc. (2007)
7. Xu K., Li W.: Exact Phase Transitions in Random Constraint Satisfaction Problems. J. Artif. Intell. Res. 12, 93--103 (2000)