Some Remarks on the Incompressibility of Width-Parameterized SAT Instances

Bangsheng Tang

Institute for Interdisciplinary Information Sciences,
Tsinghua University, 100084, Beijing
bangsheng.tang@gmail.com

Abstract. Compressibility of a formula regards reducing the length of the input, or some other parameter, while preserving the solution. Any 3-$\mathbf{SAT}$ instance on N variables can be represented by $O(N^3)$ bits; [4] proved that the instance length in general cannot be compressed to $O(N^{3-\epsilon})$ bits under the assumption $\mathbf{NP} \not\subseteq \mathbf{coNP/poly}$, which implies that the polynomial hierarchy does not collapse. This note initiates research on compressibility of $\mathbf{SAT}$ instances parameterized by width parameters, such as tree-width or path-width. Let $\mathbf{SAT}_{\mathsf{tw}}(w(n))$ be the satisfiability instances of length n that are given together with a tree-decomposition of width $O(w(n))$, and similarly let $\mathbf{SAT}_{\mathsf{pw}}(w(n))$ be instances with a path-decomposition of width $O(w(n))$. Applying simple techniques and observations, we prove conditional incompressibility for both instance length and width parameters: (i) under the exponential time hypothesis, given an instance ϕ of $\mathbf{SAT}_{\mathsf{tw}}(w(n))$ it is impossible to find within polynomial time a ϕ' that is satisfiable if and only if ϕ is satisfiable and tree-width of ϕ' is half of ϕ; and (ii) assuming a scaled version of $\mathbf{NP} \not\subseteq \mathbf{coNP/poly}$, any 3-$\mathbf{SAT}_{\mathsf{pw}}(w(n))$ instance of N variables cannot be compressed to $O(N^{1-\epsilon})$ bits.

1 Introduction

Satisfiability($\mathbf{SAT}$) is the problem of deciding whether a given conjunctive normal form($\mathbf{CNF}$) formula is satisfiable. Denote by n the input length of the formula, and by N the number of variables. $\mathbf{SAT}$ has been playing a central role in both theoretical and applied aspects of computer science. It is the prototypical $\mathbf{NP}$-complete problem in complexity theory, and has found enormous applications in various practical areas, e.g. artificial intelligence, machine learning, decision making, automated theorem proving.

Although it is not possible to solve $\mathbf{SAT}$ in its most general form within polynomial time unless $\mathbf{NP} \neq \mathbf{P}$, there is an apparent need for practically efficient algorithms. Real world instances often come with structure. Parameterization is a general way of quantifying the structure. The quest for efficient algorithms for fixed parameters (*fixed-parameter tractable*) has received significant attention in the past few years. One way of parameterizing $\mathbf{SAT}$ is using width parameters (tree-width, path-width, branch-width, etc.), which are graph-theoretic parameters of a graph associated with the instance. It is shown in a series of moves, e.g. [1, 13, 12, 5, 6, 2] that width-parameterized $\mathbf{SAT}$ with parameter k can be solved time-efficiently in simultaneously $2^{O(k)}n^{O(1)}$ time and $2^{O(k)}n^{O(1)}$ space, or space-efficiently in simultaneously $2^{O(k \log n)}n^{O(1)}$ time

J. Snoeyink, P. Lu, K. Su, and L. Wang (Eds.): FAW-AAIM 2012, LNCS 7285, pp. 192–198, 2012.

and $n^{O(1)}$ space. Rather involved time-space trade-off algorithms achieving better time and space are also given in [2].

We use the term *compressibility of* **SAT** *instances* to refer the fact that the length of input instance or parameters can be reduced by an efficient algorithm, in a way that the compressed instance preserves satisfiability. Compressing input instances has been considered in [4] under the term *sparsification*. The following two-player communication game between an efficient algorithm and an oracle was considered: the first player is a verifier who has an d-**CNF** formula, and she wants to decide its satisfiability within deterministic polynomial time; the second player is an oracle with unbounded computational power but without knowing the formula beforehand. The goal of this communication game is to minimize the number of bits the verifier must communicate with the oracle, such that the verifier can decide the satisfiability of the formula. N^d bits will suffice, because this is the total number of possible clauses and the first player can send an N^d bit string, where the ith bit indicates whether the ith clause is present. In [4], it is proved that the trivial way of communication is essentially optimal in the following sense: if the satisfiability of d-**CNF** formulas can be decided by communication within $O(N^{d-\epsilon})$ bits for any $\epsilon > 0$, then $\mathbf{NP} \subseteq \text{co}\mathbf{NP/poly}$. How about **SAT** instance of bounded width parameters? At one extreme, when the width is $O(\log n)$, the verifier can compute satisfiability by herself without any communication; at another end, when the width is $\Omega(n)$, the formula is in its most general form, previous result can be applied. The technically interesting case is for the intermediate range of this width parameter.

In this note, we focus on the compressibility of **SAT** instances parameterized by two types of width parameters, i.e. tree-width and path-width. In the setting of parameterized problems, we can discuss compression of input instances as above, or compression of parameters. Note that algorithms for parameterized problems have running time and space super-polynomial in the chosen parameter times a polynomial of the input instance length. Efficient compression of parameters would indicate significant improvement in the time and space resource requirements of the state-of-the-art algorithms. We show that compressing the width-parameter by a constant factor within polynomial time is not possible under the assumption that unless exponential time hypothesis (**ETH**) fails. **ETH** states that solving **SAT** on N variables requires $2^{\Omega(N)}$ time, which is stronger than $\mathbf{P} \neq \mathbf{NP}$ and has important implications in computational complexity (see e.g. [7, 8, 10]). Let $\mathbf{NL}[r(n)]$ be the class of problems decidable by a logspace machine equipped with a read-only non-deterministic, polynomially long witness tape, where the machine can have $O(r(n))$ passes (as the head reverses) over the witness tape. We strengthen the assumption that $\mathbf{NP} \not\subseteq \mathbf{coNP}$ to $\mathbf{NL}[\omega(1)] \not\subseteq \mathbf{coNP}$. In fact, we assume further that

$$\mathbf{NL}[\omega(1)] \not\subseteq \mathbf{coNP/poly}$$

to obtain an incompressibility result of input length for width-parameterized **SAT**.

A complexity theoretic study of this assumption is interesting on its own right, and it is left for future work. Here are some indications on its validity: (i) the belief that $\mathbf{NP} \not\subseteq \mathbf{coNP}$ is because usually people think that in fact the required certificate size blows up to exponential (not merely super-polynomial) - i.e. some kind of exhaustive

enumeration is required - and (ii) given a non-uniform advice of polynomial size will not help either (in particular, an easy extension of Karp-Lipton shows that if $\mathbf{NP} \subseteq \mathbf{coNP/poly}$ the polynomial hierarchy collapses).

Results in this note are based on preliminary observations and simple techniques. Nevertheless, this work initiates the study of compressibility or sparsification of width-parameterized $\mathbf{SAT}$ instances, and makes conceptual contributions to better understanding the complexity of width-parameterized $\mathbf{SAT}$. Techniques that work for general instances fail dramatically for width-parameterized instances. Intuition and speculation on why previous techniques fail are also included in this note. We believe that improving our results is a very interesting research direction for both theory and practice.

2 Preliminary

Notation and terminology used in this note basically follow [2].

Definition 1. *Let $G = (V, E)$ be an undirected graph. A* tree decomposition *of G is a tuple (T, X), where $T = (W, F)$ is a tree, and $X = \{X_1, \cdots, X_{|W|}\}$ where $X_i \subseteq V$ s.t. (1) $\cup_{i=1}^{|W|} X_i = V$, (2) $\forall (i, j) \in E$, $\exists t \in W$, s.t. $i, j \in X_t$, and (3) $\forall i$, the set $\{t : i \in X_t\}$ forms a subtree of T.*

Each of X_i is called a bag*, the width of (T, X) is defined as $max_{t \in W} |X_t| - 1$, and the* tree-width $\mathcal{TW}(G)$ *of graph G is defined as the minimum width over all possible tree decompositions.*

When the tree decomposition $T = (W, F)$ is restricted to a path, the decomposition is called *path decomposition*, and the specific tree-width is called path-width $\mathcal{PW}(G)$.

Definition 2. *The* incidence graph G_ϕ *of a $\mathbf{SAT}$ instance ϕ is a bipartite graph, where in one side of the bipartization each node is associated with a distinct unsigned variable, and in the other each node is associated with a clause. There is an edge between a clause-node and a variable-node if and only if the variable appears in a literal of the clause.*

The tree-width of a formula ϕ is the tree-width of its incidence graph, $\mathcal{TW}(\phi) = \mathcal{TW}(G_\phi)$. When it is clear from the context we may abuse notation and write $\mathcal{TW}(\phi)$ to denote the width of a given decomposition of G_ϕ. Note that any tree-decomposition (path-decomposition) of an instance graph will have tree-width (path-width) at most N, because one can always construct a path of the number of the clauses, and put each clause into a different bag arbitrarily, and copy all the variables into all the bags. This is a valid path-decomposition, and therefore a valid tree-decomposition of width N. Without loss of generality we assume that in number of bags is upper bounded by a polynomial of the number of the clauses and variables. This is assured by a property of decompositions called *nice*. A nice one can be constructed from any decomposition efficiently (see e.g. [9, 3]).

Denote by $\mathbf{SAT}_{\mathsf{tw}}(w(n))$ the problem of deciding satisfiability of a $\mathbf{CNF}$ formula, which is given together with a tree-decomposition of width $O(w(n))$ as input, where n

is input length. $\mathbf{SAT}_{\mathsf{pw}}(w(n))$ is the path-decomposition version. 3-$\mathbf{SAT}_{\mathsf{tw}}(w(n))$ and 3-$\mathbf{SAT}_{\mathsf{pw}}(w(n))$ are the variants where the input formulas are 3-$\mathbf{CNF}$ formulas. The following lemma shows that there is no essential difference between 3-$\mathbf{SAT}_{\mathsf{pw}}(w(n))$ and $\mathbf{SAT}_{\mathsf{pw}}(w(n))$.

Lemma 1 ([11]). $\mathbf{SAT}_{\mathsf{pw}}(w(n))$ *is reducible to 3-*$\mathbf{SAT}_{\mathsf{pw}}(w(n))$*, under logspace many-to-one reductions.*

Although the width parameter remains asymptotically unchanged, the number of variables is increased in the reduction. For each clause of k literals, at most k new variables need to be introduced. Recall that we have defined the complexity class $\mathbf{NL}[r(n)]$, which is of interest because it characterizes width-parameterized $\mathbf{SAT}$.

Lemma 2 ([11]). $\mathbf{SAT}_{\mathsf{pw}}(w(n))$ *is complete for* $\mathbf{NL}[\frac{w(n)}{\log n}]$*, under logspace many-to-one reductions.*

As in the literature, denote by $\mathbf{NP/poly}$ the class of languages accepted by a nondeterministic polynomial time Turing machine with a polynomial length advice, and $\mathbf{coNP/poly}$ its complement.

3 Incompressibility

3.1 Incompressibility of Width Parameters

We start with some preliminary observations stating that no non-trivial compression can be done to reduce the width parameter. Suppose we have an $\mathbf{SAT}_{\mathsf{tw}}$ instance ϕ together with an optimal tree decomposition of width $\mathcal{TW}(\phi) = \omega(\log n)$. A *width-compression algorithm* A with *compression ratio* α: $0 < \alpha < 1$, is an algorithm satisfying the following property (*):

> A takes ϕ and the tree decomposition as input, runs in polynomial time and then outputs another instance ϕ' along with a new tree decomposition, such that ϕ is satisfiable if and only if ϕ' is satisfiable, and $\mathcal{TW}(\phi') = \alpha\mathcal{TW}(\phi)$.

Theorem 1. *No width-compression algorithm with* $\alpha = n^{-\frac{1}{n^c}}$ *(*$c > 1$ *is a constant) for* $\mathbf{SAT}_{\mathsf{tw}}$ *instances with tree-width* $\omega(\log n)$*, satisfying (*) can exist, under* $\mathbf{ETH}$.

Proof. By $\mathbf{ETH}$, deciding satisfiability of the sub-formula by picking the clauses included in a specific bag in the decomposition in general requires $2^{\Omega(\mathcal{TW}(\phi))} = 2^{\omega(\log n)} = n^{\omega(1)}$ time. This also lower bounds the running time for any algorithm deciding satisfiability of ϕ under $\mathbf{ETH}$.

Suppose for the sake of contradiction, such an algorithm A exists. If we repeatedly run A for $\log_\alpha \mathcal{TW}(\phi) = O(\log n / \log(n^{-\frac{1}{n^c}})) = O(n^c)$ times upon ϕ, we will obtain an instance $\overline{\phi}$, where $\mathcal{TW}(\overline{\phi})$ is a constant and has the same satisfiability as ϕ. Satisfiability of $\overline{\phi}$ and the transformation from ϕ to $\overline{\phi}$ can be computed in polynomial time, which in turn implies that satisfiability of ϕ can be decided in polynomial time. However, this is impossible due to the super-polynomial lower bound for running time under $\mathbf{ETH}$ given in the previous paragraph. □

If we allow the tree-width of the instances be up to linear in n, namely general $\mathbf{SAT}$ instances, the same incompressibility result holds, while $\mathbf{P} \neq \mathbf{NP}$ is sufficient for contradiction. Namely,

Proposition 1. *No width-compression algorithm with $\alpha = n^{-\frac{1}{n^c}}$ ($c > 1$ is a constant) for $\mathbf{SAT}_{\mathsf{tw}}$ instances with tree-width $\Omega(n)$, satisfying (*) can exist, assuming $\mathbf{P} \neq \mathbf{NP}$.*

The compression ratio $\alpha = n^{-\frac{1}{n^c}}$ is a slowly increasing function as n increases with upper bound 1. When n is large enough, we can actually replace α with any constant, and the following corollary holds.

Corollary 1. *No width-compression algorithm with $\alpha = \alpha_0$ (a constant, $0 < \alpha_0 < 1$), for $\mathbf{SAT}_{\mathsf{tw}}$ instances with tree-width $\omega(\log n)$, satisfying (*) exists, under $\mathbf{ETH}$.*

3.2 Incompressibility of Instance Length

Next we turn to the question of interactively "compressing" the instance length à la [4]. Let L be a language, denote $\mathbf{OR}(L)$ with k instances is the problem: given a k-tuple $(x_1, x_2, \cdots, x_k)$, deciding whether there is an x_i, s.t. $x_i \in L$. The following lemma is crucial for the proof.

Lemma 3 ([4]). *Let L be a language, with instance length n and $t : \mathbb{Z}^+ \to \mathbb{Z}^+$ be polynomially bounded s.t. the problem of $\mathbf{OR}(L)$ with $t(n)$ instances can be decided by sending $O(t(n) \log t(n))$ bits, then $L \in \mathbf{coNP}/\mathit{poly}$.*

Note that for 3-$\mathbf{CNF}$ formulas, input length n is $O(N^3)$, therefore $\log n = \Theta(\log N)$. Now we are ready to state the incompressibility result for 3-$\mathbf{SAT}_{\mathsf{pw}}(w(n))$ instances, where $w(n) = \Omega(\log n)$.

Theorem 2. *If satisfiability of every 3-$\mathbf{CNF}$ formula on N variables, with a path-decomposition of width $O(w(n))$, where n is the input length, can be decided by the verifier through communicating $O(N^{1-\epsilon})$ bits with the oracle, then $\mathbf{NL}[\frac{w(n)}{\log n}] \subseteq \mathbf{coNP}/\mathit{poly}$.*

Proof. Consider an $\mathbf{OR}(3\text{-}\mathbf{SAT}_{\mathsf{pw}}(w(n)))$ instance, with $t(n)$ 3-$\mathbf{SAT}_{\mathsf{pw}}(w(n))$ instances each with N variables can be represented by a 3-$\mathbf{SAT}_{\mathsf{pw}}(w(n))$ instance with $t(n)N$ variables, and all the instances use different variables. We choose $t(n)$ to be polynomially bounded.

Suppose the instances are ϕ_i, $\forall i$, and each has a corresponding path-decomposition $\mathcal{P}_i$, variables $v_{i,j}$, clauses $C_{i,j}$. Merely joining all the path-decompositions sequentially will impose an AND-relation. To impose an OR-relation, additional operations are required after joining. Let a be a group of variables of length $O(\log n)$ acting as a selector, namely, for a fixed i, $(a = i)$ denotes the clause with semantic meaning "a representing the binary expansion of i". Since $t(n)$ is a polynomial in n, $O(\log n)$ bits are sufficient. For each $\mathcal{P}_i$, replace each clause $C_{i,j}$ by a clause representing $(a = i) \to C_{i,j}$, or equivalently $\overline{(a = i)} \vee C_{i,j}$. To preserve the connectivity requirement of a path-decomposition, the variables of a need to be added to each bag of the joined path.

One last problem is that each newly created clause is of $O(\log n)$ variables. To obtain a 3-$\mathbf{SAT}_{\mathsf{pw}}(w(n))$ instance, we apply the reduction by Lemma 1, blowing up the number of variables by a factor of $O(\log n)$.

In the end, a 3-$\mathbf{SAT}$ instance of $O(t(n)N \log n)$ variables with path-width $O(w(n))$ is constructed. Now by hypothesis, when $t(n)$ is a large enough polynomial this instance can be decided by the verifier through communicating $O((t(n)N \log n)^{1-\epsilon}) = O(t(n) \log t(n))$ bits with the oracle. By Lemma 3, this means 3-$\mathbf{SAT}_{\mathsf{pw}}(w(n))$ is in $\mathbf{coNP/poly}$. Combining this and the characterization in Lemma 2 concludes the proof. □

The incompressibility result for width-parameterized $\mathbf{SAT}$ seems much weaker than that for general $\mathbf{SAT}$ as in [4]. There is a crucial step called *packing lemma*, failed to be applied in width-parameterized setting. The lemma describes a procedure which combines OR of $t(n)$ $\mathbf{SAT}$ instances (each of length n) into a semantically equivalent $\mathbf{SAT}$ instance, without requiring large number of variables (only $(t(n)n)^{\frac{1}{3}}$) by allowing the clauses corresponding to different original instances to share variables. However, the same technique does not work in the width-parameterized setting since the same procedure did not take width into consideration and actually will blow up the width of resulting instance to n. Therefore, a straightforward way of combining was used in the proof of Theorem 2 which in turn requires $t(n)n$ variables. One direction of improving the result will be finding a new packing technique for width-parameterized settings.

4 Conclusion

In this note, we proved two incompressibility results, one for width parameters, the other for instance lengths. Our techniques are elementary, and future improvements with new techniques tailored for width-parameterized $\mathbf{SAT}$ are left for future work.

Acknowledgments. This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61061130540, 61073174. The author would like to thank Periklis Papakonstantinou for supervising this research.

References

[1] Alekhnovich, M., Razborov, A.A.: Satisfiability, branch-width and Tseitin tautologies. In: Foundations of Computer Science (FOCS), pp. 593–603. IEEE (2002)
[2] Allender, E., Chen, S., Lou, T., Papakonstantinou, P., Tang, B.: Width-parameterized sat: Time-space tradeoffs (2012) (manuscript)
[3] Bodlaender, H.L.: A partial k-arboretum of graphs with bounded treewidth. Theoretical Computer Science 209(1-2), 1–45 (1998)
[4] Dell, H., van Melkebeek, D.: Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In: Symposium on Theory of Computing (STOC), pp. 251–260. ACM (2010)
[5] Fischer, E., Makowsky, J.A., Ravve, E.V.: Counting truth assignments of formulas of bounded tree-width or clique-width. Discrete Applied Mathematics 156(4), 511–529 (2008)

[6] Georgiou, K., Papakonstantinou, P.A.: Complexity and Algorithms for Well-Structured *k*-SAT Instances. In: Kleine Büning, H., Zhao, X. (eds.) SAT 2008. LNCS, vol. 4996, pp. 105–118. Springer, Heidelberg (2008)
[7] Impagliazzo, R., Paturi, R.: Complexity of k-sat. In: Proceedings of Fourteenth Annual IEEE Conference on Computational Complexity, pp. 237–240. IEEE (1999)
[8] Impagliazzo, R., Paturi, R., Zane, F.: Which problems have strongly exponential complexity? Journal of Computer and System Sciences (JCSS) 63(4), 512–530 (2001); (also FOCS 1998)
[9] Kloks, T.: Treewidth: computations and approximations, vol. 842. Springer (1994)
[10] Lokshtanov, D., Marx, D., Saurabh, S.: Known algorithms on graphs of bounded treewidth are probably optimal. In: 22nd ACM/SIAM Symposium on Discrete Algorithms (SODA 2011), pp. 777–789 (2011)
[11] Papakonstantinou, P.A.: A note on width-parameterized sat: An exact machine-model characterization. Information Processing Letters (IPL) 110(1), 8–12 (2009)
[12] Samer, M., Szeider, S.: A fixed-parameter algorithm for# sat with parameter incidence treewidth. Arxiv preprint cs/0610174 (2006)
[13] Szeider, S.: On Fixed-Parameter Tractable Parameterizations of SAT. In: Giunchiglia, E., Tacchella, A. (eds.) SAT 2003. LNCS, vol. 2919, pp. 188–202. Springer, Heidelberg (2004)