Information Processing Letters 135 (2018) 62–67

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Cliques enumeration and tree-like resolution proofs

Massimo Lauria

Dipartimento di Scienze Statistiche – Sapienza Università di Roma, Italy

ARTICLE INFO

Article history:
Received 22 November 2017
Received in revised form 20 February 2018
Accepted 1 March 2018
Available online 6 March 2018
Communicated by Marcin Pilipczuk

Keywords:
Theory of computation
Resolution
Clique
Decision tree

ABSTRACT

We show the close connection between the enumeration of cliques in a k-clique free graph G, the running time of DPLL-style algorithms for k-clique problem, and the length of tree-like resolution refutations for formula Clique(G,k), which claims that G has a k-clique. The length of any such tree-like refutation is within a "fixed parameter tractable" factor from the number of cliques in the graph. We then proceed to drastically simplify the proofs of the lower bounds for the length of tree-like resolution refutations of Clique(G,k) shown in Beyersdorff et at. 2013, Lauria et al. 2017, which now reduce to a simple estimate of said quantity.

1. Introduction

The k-clique problem asks whether a graph has a set of k pairwise connected vertices, i.e., a *k-clique*. Being one of the standard NP-complete problems it seems very unlikely that it has an efficient algorithm, even in approximation [26,24]. The brute force approach to solve the problem on a graph of n vertices is to check all of the $\approx n^k$ sets of k vertices. Even more sophisticated algorithms (for example [38]) still run in time $n^{\Omega(k)}$ in the worst case. Is this the best we can hope for? If one believes the Exponential Time Hypothesis we cannot even get down to $n^{o(k)}$ [25,29]. The problem is so difficult that we would like to prove its hardness without using unproved hypotheses. Unfortunately we cannot do that unless we restrict the computational model. For example we know that circuits that solve k-clique must have size $n^{\Omega(k)}$ if they are restricted to be either of constant depth [36] or without negation gates [35]. These results hold even if we are satisfied with solving k-clique only asymptotically almost surely in the Erdős–Renyi model, where the edges of the graph are picked (independently) at random according to the appropriate density.

In this paper we focus on *decision trees* and DPLL-style algorithms [16,15] for the k-clique problem, namely algorithms that explore the space of solutions by guessing possible members for the k-clique. In case a choice happens not to be fruitful, the algorithm backtracks and tries other possibilities. The execution of a DPLL-style algorithm is represented by a decision tree, therefore identify the two concepts.

Given a graph G with no k-cliques, we can build a propositional formula Clique(G,k) that falsely claims that G has a clique of size k. A correct algorithm that searches for a k-clique in G must fail, and its trace be an efficiently verifiable proof that Clique(G,k) is unsatisfiable. When this algorithm is simple enough, as in the case of decision trees, the proof can be written down in a simple language as well. If such language does not allow for short proofs of unsatisfiability of Clique(G,k), the running time of the algorithm must be long too. Decisions trees produce, on failure, *tree-like resolution* proofs of unsatisfiability of formula Clique(G,k), for which we can prove strong lower bounds. By studying how the proof is written down, we can ignore how the algorithm finds it. A lot of those details can be abstracted away up to the point that we can see the

E-mail address: massimo.lauria@uniroma1.it.

https://doi.org/10.1016/j.ipl.2018.03.001
0020-0190/

proof as the result of a non-deterministic, i.e., "very lucky", proof search. Concretely, it means that a lower bound for the length of tree-like resolution proofs applies to all decision trees, regardless of how smart their decision strategy is.

The field of *proof complexity* [14,6,37] studies the length of proofs of propositional unsatisfiability. The languages in which such proofs are written down are called *proof systems*. The more general the proof system, more general is the class of algorithms that the system captures. Resolution is definitely the most famous and it is at the core of state-of-the-art SAT algorithms [4,30,31]. The first proof length lower bound for resolution was proven in [23], followed by more general lower bound techniques [27,8]. In practical SAT solving memory is a resource as precious as time, hence [19,1] developed a notion of proof space that is supposed to model memory usage. Another important proof complexity parameter of resolution proofs is the "width", i.e., the size of each proof line. Estimating the required width of a resolution proof is a proxy to estimate the required length [8] or the required space [3,21]. The correspondence between width and space is not very tight [33], which leads to study resolution space directly, using various pebbling games to model memory allocation. While it is hard to know how much memory is needed to win these games in general [22,13], it is possible to use and combine well understood cases of these games in order to prove resolution space lower bounds and resolution length vs space trade-offs [32].

We now go back to the problem of determining the resolution proof complexity of Clique(G,k). For $k \approx n$, super polynomial lower bounds have been proved using the size-width relation [5,8], which is by now a standard technique in proof complexity. For $k \ll n$ the problem is still open. Neither the size-width relation nor interpolation [27], which is the other main tool used in literature to prove resolution lower bounds, give anything for $k \ll n$. New techniques may be necessary to solve this problem.

In this paper we focus on the *tree-like resolution*, which is a restricted form of resolution that captures algorithms based on decision trees. Tree-like resolution is weaker than general resolution [12], but we understand better its proof length and space [9,20]. There are three types of graphs for which the clique formula is known to require tree-like resolution refutations of length $n^{\Omega(k)}$. In [10] the lower bound is proved for the complete $(k-1)$-partite graph, as well as for Erdős–Renyi random graph with appropriate edge density. A lower bound of $n^{\Omega(\log(n))}$ holds for Ramsey graphs, i.e., graphs of n vertices that have neither a $2\log(n)$-clique nor a $2\log(n)$ independent set [28]. In this paper we do not improve on these results, but we drastically simplify them by showing a connection between the length of tree-like resolution refutations of Clique(G,k) and the number of cliques in G.

The paper is organized as follows. In Section 2 we give the necessary definitions and notations. In Section 3 we show the close correspondence between the number of cliques in a graph and the length of tree-like resolution refutations for the clique formulas on that graph. In Section 4 we show classes of k-clique free graphs with many cliques, hence they need large refutations for the corresponding clique formulas.

2. Preliminaries

In this paper we consider simple undirected loop-less graphs $G = (V, E)$. We write $\Gamma(v)$ to denote the set of vertices in V that are neighbors of a vertex $v \in V$. For an arbitrary set of vertices $U \subseteq V$ we denote as $\Gamma(U)$ the set of vertices $\bigcap_{u\in U} \Gamma(u)$, i.e., the common neighbors of U in G. A clique of G is a set of vertices so that there is an edge between any two of them. We denote as $\mathcal{C}(G)$ the set of cliques of G. We denote as $[m]$ the set of integers from 1 to m.

A CNF formula over a set of variables is a conjunction of distinct clauses, each of them being the disjunction of distinct literals. A literal is either an occurrence of a variable or its negation. D is a subclause of a clause C when D is a disjunction of literals contained in C. We indicate that D is a subclause of C with notation $C \subseteq D$.

The *k-clique formula* Clique(G,k) is a CNF formula over variables $x_{i,v}$ for every $v \in V$ and $i \in [k]$, where the boolean variable $x_{i,v}$ indicates whether the i-th vertex of the clique is v. The formula is the conjunction of clauses

$$\bigvee_{v\in V} x_{i,v} \qquad \forall i \in [k]\,, \tag{1a}$$

$$\neg x_{i,u} \vee \neg x_{j,v} \qquad \forall i, j \in [k], i \neq j, \forall u, v \in V, \{u, v\} \notin E\,, \tag{1b}$$

$$\neg x_{i,u} \vee \neg x_{i,v} \qquad \forall i \in [k], \forall u, v \in V, u \neq v\,. \tag{1c}$$

Clauses (1a) are called clique axioms, clauses (1b) are called edge axioms, and clauses (1c) are called functionality axioms. Clearly Clique(G,k) is satisfiable if and only if G contains a clique of k vertices, and this holds even without the functionality axioms.

Tree-like resolution and Decision trees. The proofs of unsatisfiability of Clique(G,k) formula that we consider in the paper are sequences of logical inference steps obtained using the *resolution rule*, an inference rule that derives a clause from two clauses, called premises, as follows

$$\frac{A \vee x \qquad B \vee \neg x}{A \vee B}\,. \tag{2}$$

To apply rule (2) the two premises must contain, respectively, the positive and negative literal of some variable x, and thus we say that we *resolve* the two clauses over variables x. The derived clause is their *resolvant*, and it is true whenever both premises are true.

A *tree-like resolution proof* of a clause C from some CNF formula F is a rooted binary tree, directed from the leaves to the root, where each node in the tree is labeled by a clause over the variables of F. The clauses labeling the nodes in the proof must have the following properties:

- no clause contains both a literal and its negation;
- the clause labeling an internal node is the resolvant of the clauses labeling its two immediate predecessors;
- the clause at the root is a subclause of C;