The Computational Complexity Column

by

Jacobo Torán

Dept. Theoretische Informatik, Universität Ulm
Oberer Eselsberg, 89069 Ulm, Germany
toran@informatik.uni-ulm.de
http://theorie.informatik.uni-ulm.de/Personen/jt.html

Starting with this issue I take over the position of Lance Fortnow as editor of this column. First of all I would like to thank him for his excellent work. He has been able to touch in this column many different topics, opening the area and reflecting how broad the field of computational complexity really is. You can still follow regularly Lance's views on complexity theory on his very informative Web Log under http://www.fortnow.com/lance/complog/

I will continue with Lance's editorial policy and publish in the column survey articles on recent developments in the different complexity areas. My first column is devoted to proof complexity, a very active area of research located at the intersection of complexity and logic.

Space and Width in Propositional Resolution

Jacobo Torán

Abstract

Resolution is, due to its simplicity and its relation to several automated theorem proving algorithms, one of the best studied propositional proof systems. The most important complexity measure of a

resolution proof is its size, the number of clauses used in the proof. In the last years, in an effort to better understand this proof system, other complexity measures for resolution have been introduced. We focus this survey on the known results about two of these complexity measures: space and width. We review the relationship between these measures and the size in resolution, mentioning tradeoff results as well as characterizations of width and space in terms of combinatorial games.

1 Introduction

The main motivation for studying the complexity of proving tautologies or refuting unsatisfiable formulas in concrete systems comes from the close relationship between these problems and central questions in complexity theory. For example, Cook and Reckhow [12] proved the equivalence of separating the complexity class NP from coNP and the question of obtaining superpolynomial lower bounds for the refutation of a family of unsatisfiable formulas in every propositional proof system. Since it is very hard to reason about arbitrary proof systems, research in the area has focused on the study of simple concrete systems. This is similar to the development of circuit complexity, where advances in the field come from the study of concrete circuit classes. Perhaps the simplest non-trivial propositional proof system is resolution [27]. It has a single derivation rule and works only with clauses. This simplicity together with the fact that several automated theorem proving algorithms used in practice are based on resolution has provided the motivation for important research on this proof method. Resolution is probably the best studied propositional proof system although the power of the method is not yet completely understood.

The most important complexity measure for resolution is the *size* or the number of clauses used in a refutation. This corresponds to the concept of time in an algorithm. During the last decades some of the most influential research in the area of proof complexity has proved exponential lower bounds for the resolution size of important classes of unsatisfiable formulas [20, 31, 10, 3]. Recently in an effort to unify some of the existing results and to gain a better understanding of this proof system other complexity measures for resolution have been introduced. These measures are related to the notion of memory consumption or space used by an algorithm. We focus this survey on two of these complexity measures: width and space.

The notion of resolution *width* was first made explicit by Galil [19] but the importance of the concept was pointed out more recently by Ben-Sasson and

Wigderson in [7]. The width of a resolution refutation is the largest number of literals in a clause used in the refutation. Ben-Sasson and Wigderson proved a connection between the minimal width of a resolution refutation for a formula and the size of the refutation, thus providing a new method for proving lower bounds on resolution size.

Esteban and Torán introduced in [17] the concept of resolution *space*, transforming a previous definition from [21]. The space of a resolution refutation is the number of clauses that have to be kept simultaneously in memory to infer a contradiction. As in the definition of space in a Turing machine, the space used by the input clauses is not considered in the model and these clauses can be downloaded to the working memory when needed. Alekhnovich, Ben-Sasson, Razborov and Wigderson [1] extended the definition of resolution space in several ways. They defined a space notion for stronger proof systems like Propositional Calculus, and introduced finer space measures like the *variable space* or the *bit space*, measuring respectively the occurrences of variables and the bit size of the clauses that are simultaneously kept in memory.

In recent years several results obtaining non-trivial upper and lower bounds for the different resolution complexity measures have been published. We review here some of these results and show that in spite of its very different nature the three measures of size, width and space are tightly interrelated. We also review the existing tradeoffs in the resources as well as purely combinatorial characterizations of the concepts of resolution width and tree-space in terms of 2-person games. These beautiful characterizations stress the implicitness of these measures as purely combinatorial properties of the input formulas, independent of the notion of resolution.

2 Definitions

We will consider formulas over a set of propositional variables V. A *literal* is either a variable $x \in V$ or its negation $\overline{x}$. A *clause* is a disjunction of literals and a formula F in conjunctive normal form (CNF) is a conjunction of clauses. For a literal l and $a \in \{0,1\}$, $F_{l=a}$ represents the formula obtained from F by giving to l the value a and reducing the formula in the intuitive way.

Resolution is a refutation proof system for CNF formulas. The only inference rule in this proof system is the resolution rule:

$$\frac{C \vee x \qquad D \vee \bar{x}}{C \vee D} .$$

Cutting variable x from clauses $C \vee x$ and $D \vee \bar{x}$ one gets the *resolvent* clause $C \vee D$. A resolution refutation of a CNF formula F is a sequence of clauses $C_1, \ldots, C_s$ where each C_i is either a clause from F or is inferred from earlier clauses by the resolution rule, and C_s is the empty clause. We will denote the empty clause by $\Box$. Resolution is sound and complete, which means that such a refutation for a formula exists if and only if F is unsatisfiable. A resolution refutation can be seen as a directed acyclic graph, in which the clauses are the vertices, and if two clauses are resolved then there is a directed edge going from each of the two clauses to the resolvent. If the underlying graph in a refutation happens to be a tree, we talk about *tree-like* resolution. It is known that for certain formulas general resolution can produce exponentially shorter refutations than tree-like resolution [9, 6]. The reason for this is that, contrary to general resolution, in a tree-like proof a clause that is needed more than once in the refutation must be re-derived each time from the initial clauses.

The *size* of a resolution refutation is the number of clauses it contains. For an unsatisfiable formula F, size(F) denotes the minimal size of a resolution refutation of F. We denote by tree-size(F) the minimal size of a tree-like resolution refutation of F. As mentioned above, this can be greater than size(F). We define now resolution width and space, the other two complexity measures discussed in this survey.

Definition 1. *[7] The width of a clause is the number of literals appearing in it. For a set of clauses $\mathcal{C}$ ($\mathcal{C}$ can be for example a formula in CNF or a resolution refutation) the width of $\mathcal{C}$, denoted by initial-width($\mathcal{C}$), is the maximal width of a clause in the set $\mathcal{C}$.*

The width needed for the resolution of an unsatisfiable CNF formula F, denoted by width(F), is the minimal width needed in a resolution of F, that is, the minimum of initial-width(π) over all resolution refutations π of F.

The space needed in a resolution refutation is the number of clauses that have to be kept simultaneously in memory (not counting the initial clauses) in order to derive the empty clause. More formally:

Definition 2. *[17, 1] For $k \in N$, we say that an unsatisfiable CNF formula F has resolution refutation bounded by space k if there is a sequence of CNF formulas $F_1, \ldots, F_s$, such that $F_1 \subseteq F$, $\Box \in F_s$, in any F_i there are at most k clauses, and for each $i < s$, F_{i+1} is obtained from F_i by deleting some of its clauses, adding the resolvent of two clauses of F_i, or adding one of the clauses of F (initial clause).*

The space needed for the resolution of an unsatisfiable formula F, denoted space(F) is the minimum k for which the formula has a refutation bounded by space k.

In [1] the authors introduce two refinements of the space measure: the variable space, and the bit space. For an unsatisfiable formula F, the variable space of F, Vspace(F) is defined like the space of F above, but counting the sum of sizes (widths) of the clauses of the formulas F_i in the resolution, instead of just the number of clauses. The bit space counts the length (as words over a finite alphabet) of the clauses kept in memory.

3 Width

The idea that a large resolution refutation must contain wide clauses has been around for many years. Already the first proof of an exponential lower bound for the resolution size given by Haken [20] was based on the fact that any resolution refutation for the family of formulas encoding the pigeon hole principle must contain many wide clauses of a certain kind. The use of wide clauses to prove resolution lower bounds was also an important ingredient in subsequent results, especially in [3]. In [19] it was proved that the refutation of certain formulas require large width and it was conjectured in [22] that a formula having only resolution refutations with large clauses cannot have short refutations. But the exact relation between width and size in resolution was first explicitly stated by Ben-Sasson and Wigderson in [7]. Expressing the relation between size and width they reduced the problem of giving lower bounds on the size of a refutation to that of giving lower bounds on the width. The size-width relations for resolution were inspired by similar bounds regarding relations between the degree in the Polynomial Calculus proof system and the resolution size obtained in [11]. With this method Ben-Sasson and Wigderson were able to provide a unified approach to prove most of the previously known lower bounds for resolution size. They also used the width concept to develop a simple algorithm for searching for a refutation for a given formula F. The algorithm searches systematically for clauses of increasing size and works in time $n^{O(w)}$ where n is the number of variables in F and w the minimal width of any refutation of F.

For the proof of the width-size relation the following technical lemma is needed:

Lemma 3. *Let F be a formula in CNF, l be a literal in F and let F^l denote the set of clauses in F containing literal l. If width$(F_{l=1}) \leq k-1$ and width$(F_{l=0}) \leq k$ then width$(F) \leq \max\{k,$ initial-width$(F^l)\}$.*

Proof. The set of clauses $F_{l=1}$ is obtained from F by removing literal $\overline{l}$ from all the clauses in F and deleting the clauses containing literal l. By hypothesis, there is a width $k-1$ resolution refutation π of $F_{l=1}$. Every

initial clause in $F_{l=1}$ is either an initial clause from F or a clause obtained by deleting literal $\bar{l}$ from a clause in F. By adding $\bar{l}$ to the initial clauses of this second type in $F_{l=1}$ and propagating this literal through π we obtain a legal resolution derivation π' of $\bar{l}$ from F of width at most k. In order to derive the empty clause, now $\bar{l}$ can be resolved with all the clauses in F^l obtaining $F_{l=0}$. This part has width initial-width(F^l). By the assumption there is a resolution refutation of $F_{l=0}$ with width $\leq k$. The width of the whole refutation is therefore $\max\{k$,initial-width$(F^l)\}$. ■ We can now state the basic relation between the complexity measures of resolution width and size:

Theorem 4. *[7] For an unsatisfiable formula F in CNF with n variables*

$$width(F) \leq \ initial\text{-}width(F) + O(\sqrt{n \ln(size(F))}).$$

Proof. Let F be a formula in CNF with n variables and let k be its initial width. Let π be a resolution refutation of minimal size s. We define d and a to be $d := \lceil \sqrt{2n \ln s} \rceil$ and $a := (1 - \frac{d}{2n})^{-1}$. A clause in π is called *fat* if it has more than d literals. Let π^* be the set of fat clauses in π. We prove by induction on n that width$(F) \leq d + k + \log_a(|\pi^*|)$. The result follows from this implication since $|\pi^*| \leq s$ and therefore by the way a is and d are defined, $\log_a(|\pi^*|)$ is bounded by $c\sqrt{2n \ln s}$ for some constant c. The base case $n = 0$ holds trivially. For the induction case, observe that F contains at most $2n$ literals and therefore one literal l appears in at least $\frac{d}{2n}|\pi^*|$ fat clauses. We consider the two refutations of the formulas $F_{l=0}$ and $F_{l=1}$ obtained from π by setting literal l to 1 and to 0 respectively. Setting $l = 1$ removes all the clauses including literal l and leaves a refutation of $F_{l=1}$ with at most $(1 - \frac{d}{2n})|\pi^*| = a^{-1}|\pi^*|$ fat clauses. By induction hypothesis we have width$(F_{l=1}) \leq d + k + \log_a(a^{-1}|\pi^*|) = d + k + \log_a(|\pi^*|) - 1$. Setting $l = 0$ produces a refutation of the formula $F_{l=0}$ with less than n variables, and again by induction on n it holds width$(F_{l=0}) \leq d + k + \log_a(|\pi^*|)$. The result is obtained by applying Lemma 3. ■

From this result it follows that lower bounds for the resolution width can imply lower bounds for the resolution size.

Corollary 5. $size(F) = exp(\Omega(\frac{(width(F) - initial\text{-}width(F))^2}{n}))$.

For the case of tree-like resolution refutations, Ben-Sasson and Wigderson obtain the relation:

Theorem 6. $tree\text{-}size(F) \geq 2^{(initial\text{-}width(F) - width(F))}$.

Observe that in the exponent of the right hand side of both relations we have the difference between the minimal resolution width and the initial width of F. Because of this fact, for the case of families of formulas with large initial clauses, this method can prove only trivial size lower bounds. For formulas with constant initial width (formulas in 3-CNF for example) Theorem 4 implies a superpolynomial lower bound in the resolution size in case one can prove a width lower bound of $\Omega(\sqrt{n \ln n})$ where n is the number of variables in the formula. Bonet and Galesi [8] have shown that this bound is basically optimal by giving a family of 3-CNF formulas MGT_n, built over $O(n^2)$ variables, having polynomial size resolution refutations, but requiring $\Omega(n)$ width (the square root of the number of variables) to be refuted. MGT_n is the negation of a graph tautology encoding the principle that a directed acyclic graph closed under transitivity must have a source node. For the case of tree-like refutations, the optimality of the width-size relation stated in Theorem 6 was proved already in [7].

Ben-Sasson and Wigderson use the size-width relation to provide a simplified and unifying proof of the existing lower bounds for the resolution size of random k-CNF formulas, Tseitin formulas [1] and formulas for the pigeonhole principle.

Another application of the width concept in resolution is the development of a dynamic algorithm for searching for a resolution refutation of an unsatisfiable formula F. The procedure seeks in a systematic way for a minimal width refutation for F obtaining systematically for increasing values of w all the clauses of width $\leq w$ that can be derived from the initial set of clauses and adding them to this set. It is not hard to see that the running time of this algorithm is bounded by $n^{O(\text{width}(F))}$, where n is the number of variables in F. A similar proof search algorithm with slightly better parameters was given by Beame and Pitassi in [3]. Ben-Sasson and Wigderson [7] prove that the dynamic algorithm never performs much worse than standard recursive methods used in practice such as the Davis-Putnam procedures [14]. They also provide a family of formulas for which the above explained width-based algorithm works exponentially faster.

4 Space

As explained in the introduction, the resolution space of a formula is the minimum number of clauses that have to be kept simultaneously in memory in order to refute the formula. There is, however, a more natural way

[1] These formulas were defined by Tseitin in [30] and express the principle that the sum of the degrees of the vertices in a graph must be even.

to look at the space definition using the pebble game on graphs, a traditional model used for space measures in complexity theory and for register allocation problems (see [28]). We can give an orientation to the graph representation of a resolution refutation having the initial clauses as sources and the empty clause as the unique sink. It was observed in [17] that the space required for the resolution refutation of a CNF formula F (as expressed in Definition 2) corresponds to the minimum number of pebbles needed in the following pebble game played on the graph of a refutation of F.

Definition 7. *Given a connected directed acyclic graph with one sink the aim of the game is to put a pebble on the sink of the graph following this set of rules:*

1) A pebble can be placed on any initial node, that is, on a node with no predecessors.

2) A pebble can be removed from any node.

3) A pebble can be placed on an internal node provided all its parent nodes are pebbled. In this case, instead of placing a new pebble on it, one can shift a pebble from a parent to the node.

Lemma 8. *[17] Let F be an unsatisfiable CNF formula. space (F) coincides with the minimum number of pebbles needed for the pebble game played on the graph of a resolution refutation of F.*

An upper bound for the resolution space of an unsatisfiable formula F with n variables is $n+1$. This is so because every such formula can be refuted by a tree-like resolution of depth at most n, and it is a well known fact that a binary tree of depth n can be pebbled with $n+1$ pebbles.

The formula on n variables containing all 2^n possible clauses including the n variables needs resolution space $n+1$ [17]. This bound is the worse possible in terms of variables but is only logarithmic with respect to the number of clauses. In [29] it was shown that the family T_n of Tseitin formulas associated to certain expander graphs have $4n$ variables $256n$ clauses and require space $n-3$ for every n. The pigeonhole formulas $\neg PHP_n^m$ expressing the principle that m pigeons do not fit in n holes for $m > n$, have resolution space exactly $n+1$ independently of the number of holes [29]. A general method for proving space lower bounds that unifies these results is shown in [1].

Ben-Sasson and Galesi prove in [5] a lower bound of $\Omega(n/\Delta^{1+\epsilon})$ for the space needed in the resolution of unsatisfiable random k-CNF formulas over n variables and Δn clauses. An upper bound of $O(n\Delta^{-\frac{1}{k-2}})$ for the resolution space of the same class of formulas is given by Zito in [32]. A thorough

exposition of the existing upper and lower bounds for resolution space is contained in [15].

The characterization of space of a resolution refutation in terms of the pebble game played on the refutation makes it possible to measure the tree-like resolution space for a formula F as the minimum number of pebbles needed for the game on a tree-like refutation of F. As mentioned before, for certain classes of formulas, there can be an exponential gap in the sizes of tree-like and general refutations. Observe that the width measure does not increase when transforming a general resolution refutation into a tree-like one since the clauses that might be rederived do not increase the width of the refutation. For the case of space, it is shown in [18] that for certain formulas there can be a linear separation between the space measure in tree-like and general refutations.

It is not clear that space lower bounds can imply size lower bounds as in the case of the width measure. The next result shows that this holds, however, for the case of tree-like resolution size.

Theorem 9. *[17] For an unsatisfiable formula F in CNF*

$$\textit{tree-size}(F) \geq 2^{\textit{space}(F)} - 1.$$

Proof. We will show that the resolution tree in the refutation of F can be pebbled with $d+1$ pebbles, where d is the depth of the biggest complete binary tree embedded[2] in the refutation graph. As the biggest possible complete binary tree embedded in a tree of size s has depth $\lceil \log s \rceil$, the theorem holds. It is a well known fact that $d+1$ pebbles suffice to pebble a complete binary tree of depth d (with the directed edges pointing to the root). In fact $d+1$ pebbles suffice to pebble any binary tree whose biggest embedded complete binary tree has depth d. In order to see this we use induction on the size of the tree. The base case is obvious. Let T be refutation tree, and T_1 and T_2 be the two subtrees from the root. Let us call $d_c(T)$ the depth of the biggest embedded complete subtree in T.

$$d_c(T) = \begin{cases} \max(d_c(T_1), d_c(T_2)) & \text{if } d_c(T_1) \neq d_c(T_2) \\ d_c(T_1) + 1 & \text{if } d_c(T_1) = d_c(T_2) \end{cases}$$

By the induction hypothesis one can pebble T_1 with $d_c(T_1)+1$ pebbles and T_2 with $d_c(T_2)+1$ pebbles. Let us suppose that $d_c(T_1) < d_c(T_2)$, then $d_c(T) = d_c(T_2)$ and one can pebble first T_2 with $d_c(T_2)+1$ pebbles, leave a pebble in the root of T_2 and then pebble T_1 with $d_c(T_1)+1$ pebbles. For this

[2] A tree T' is embedded in T if T can be obtained from T' by adding nodes and edges or inserting nodes in the middle of edges of T.

second part of the pebbling one needs $d_c(T_1) + 2 \leq d_c(T_2) + 1$. The other case is similar. ■

From this result follows that a lower bound for the resolution space of a formula implies an exponentially larger tree-like resolution size lower bound. An upper bound for the size in terms of the space for the case of general resolution is given in [17], but the height measure is also considered as a parameter in the relation. The height of a resolution refutation is the length of the longest path from an initial clause to the empty clause in the refutation graph.

Theorem 10. *[17] For an unsatisfiable formula F in CNF*

$$size(F) \leq \binom{space(F) + height(F)}{space(F)}.$$

A consequence of this result is that refutations with polynomial height and constant space must have polynomial size.

5 Tradeoffs

Ben-Sasson has shown in [4] tradeoff results for the size, width and space complexity measures applied to certain families of formulas. He proves that for these formulas there are optimal resolution refutations with respect to one of the parameters, but the optimization of one complexity measure increases the others. A central part in the proof of this result is a surprising connection between the variable space needed in the resolution of certain formulas and the black-white pebbling measure of an underlying graph. This connection is stated here without proof in Lemma 13. The black-white pebbling measure of a circuit G introduced in [13] intuitively measures the space needed for the simulation of the circuit on a nondeterministic Turing machine. This game is more involved than the standard pebbling game and includes pebbles of two kinds: white pebbles simulating nondeterministic steps and black ones for the simulation of deterministic steps. We omit here the details about this game and refer the interested reader to [25].

The formulas used by Ben-Sasson to prove the tradeoff behavior are the pebbling contradictions introduced in [7]. They express the principle that in a directed acyclic graph, pebbling the source nodes and following the rule that if all the predecessors of a node v contain a pebble then v also gets one, implies that a pebble will be placed on the sink.

Definition 11. *Let $G = (V, E)$ be a directed acyclic graph in which every vertex has fan-in 2 or 0 with a unique sink s. We call a graph with these*

properties a circuit-graph. Let $k > 0$. We associate k distinct Boolean variables $v_1, \ldots, v_k$ with every vertex $v \in V$. Peb_G^k, the k-th degree pebbling contradiction of G, is the conjunction of:

Source axioms: $\bigvee_{i=1}^{k} v_i$ *for each source* v.

Pebbling axioms: $\overline{u}_i \vee \overline{v}_j \vee \bigvee_{l=1}^{k} w_l$ *for* u *and* v *the two predecessors of* w, $i, j \in \{1 \ldots k\}$.

Sink axioms: $\overline{s}_i$ *for the sink* s *and* $i \in \{1 \ldots k\}$.

Ben-Sasson shows that for certain circuit-graphs G the tree-like resolution of the formulas Peb_G^1 presents the following tradeoff:

Theorem 12. *[4] For infinitely many integers n, there exist contradictions F_n of size n such that:*

1: F_n has tree-like resolution refutations of linear size and constant space.

2: $width(F_n) = O(1)$.

3: For any tree-like refutation π of F_n: $width(\pi) \cdot \log(size(\pi)) = \Omega(\frac{n}{\log n})$.

4: For any tree-like refutation π of F_n: $width(\pi) \cdot space(\pi) = \Omega(\frac{n}{\log n})$.

Proof. For proving the first part of the result, consider a circuit-graph G with n vertices and define a topological order on its vertices. In the formula Peb_G^1 there is exactly one variable associated to each vertex in G and therefore we can identify the two. We describe a tree-like resolution refutation with the desired properties. Starting with the sink clause $\overline{s}$, we keep an active clause C containing only negative literals that is always resolved with an initial clause. In each resolution step we consider the variable v in C corresponding to the highest node in the clause in the topological ordering and resolve C with the unique initial clause in Peb_G^1 containing the positive literal v. This can be the clause v in case the node v is a source, or the clause $\overline{u} \vee \overline{w} \vee v$ in case v is an internal node with predecessors u and w. In both cases variable v is resolved and the new active clause C only contains negative literals. Observe that once a variable v is resolved from the active clause, it will never appear there again since the variables in this clause will always be smaller than v in the topological order. Since in every resolution step a variable is deleted from the active clause, there are as many resolution steps as variables in G. Counting the initial clauses, the size of the refutation is bounded by $2n$. The underlying resolution graph is a very thin tree since in each step an initial clause is resolved and therefore the space needed for pebbling this tree is 2.

It is not hard to see that part 2 of the result holds for Peb_G^k for every circuit-graph G and every $k > 0$. A refutation for this formula just has to resolve the clauses following a topological order of the vertices in G starting with the lowest vertex in the order and obtaining for each vertex v the intermediate clause $\bigvee_{i=1}^{k} v_i$. It follows by induction on the topological order that this can be achieved using intermediate clauses of width at most $\max\{2k, k+2\}$.

For proving the more involved parts 3 and 4 of the theorem, Ben-Sasson shows a beautiful connection between the variable space needed for the resolution of of Peb_G^1 and the black-white pebbling measure of the underlying graph G.

Lemma 13. *[4] For any circuit-graph G*

$$Vspace(Peb_G^1) \geq \textit{ black-white pebbling number of } G.$$

It is known that for infinitely many n there are circuit-graphs G with n vertices and having black-white pebbling number $\Omega(\frac{n}{\log n})$ [24]. Let G be such a circuit-graph. By Theorem 9, if Peb_G^1 has a tree-like resolution refutation of size S, then this refutation needs space $\log(S)$. By the above Lemma, the variable space of the refutation must be at least $\frac{n}{\log n}$ and, therefore the refutation must contain some clause of width at least $\frac{n}{\log S \log n}$. This proves parts 3 and 4 of Theorem 12. ■

This result implies that tree-like resolution refutations of minimal width for the pebbling contradictions will have exponential size, and therefore proof findings methods based on searching for proofs of small width, as the one mentioned at the end of Section 3, will perform very badly on such formulas.

A result similar to Theorem 12 but for the case of general resolution is also given by Ben-Sasson in [4].

6 Combinatorial Characterizations

The complexity measures of resolution width and tree-like resolution space for unsatisfiable formulas in CNF have been exactly characterized in terms of 2-person combinatorial games played on the formulas. Besides being useful for proving new width or space bounds, these results are remarkable because they show that the two complexity measures only depend on the structure of the formulas and can be defined independently of the notion of resolution. In both cases the game giving the characterization had been previously defined in the literature for other purposes.

6.1 A Game Characterization of Resolution Width

Atserias and Dalmau [2] characterized resolution width in terms of the existential k-pebble game. This combinatorial tool had been introduced by Kolaitis and Vardi [23] in the context of Finite Model Theory. We present here a simplified version of this game which we call Game A.

Game A is played in rounds by two players, Spoiler and Duplicator [3], on an unsatisfiable formula F in CNF. Both players construct a partial assignment α of the variables in F. The goal of Spoiler is to falsify one of the initial clauses of F with the constructed assignment while keeping the partial assignment as small as possible. Duplicator tries to keep this from happening. Initially α is the empty assignment and in each round Spoiler can delete some of the values assigned in α and asks for the value of a variable x in F not assigned in α. Duplicator extends α with a Boolean value for x. The game finishes when one of the initial clauses is falsified by α. In this case we say that Duplicator scores as many points as the maximum number of variables that are assigned simultaneously in α at some moment in the game.

Definition 14. *For an unsatisfiable formula F in CNF we define $game_A(F)$ to be the maximum number of points scored by Duplicator on F against an optimal terminating strategy of Spoiler.*

Theorem 15. *[2] For an unsatisfiable formula F in CNF*

$$width(F) = game_A(F) - 1.$$

Proof. We first give a strategy for Spoiler in the game against which Duplicator scores at most $\text{width}(F) + 1$ points. Consider the graph of a resolution refutation of minimal width for F. Starting at the empty clause, the strategy of Spoiler is to simulate a path from this clause to one of the initial clauses in the refutation. Every round starts with the constructed partial assignment α falsifying a clause C in this path. Spoiler asks for the value of the variable x being resolved to generate C. By the nature of resolution any of the two possible extensions of α to x falsifies one of the parent clauses of C. Depending on the choice of Duplicator, Spoiler moves to the corresponding falsified parent clause and deletes from α the variables not appearing in the new clause. Following this strategy Spoiler eventually reaches an initial clause. Observe that the number of variables assigned simultaneously by α is at most the maximal width of a clause in the refutation plus the variable being resolved in each round.

[3] The names of the players are justified in the context of Finite Model Theory

In order to prove that $\text{width}(F) \leq \text{game}_A(F) - 1$ we give a strategy for Duplicator that never falsifies an initial clause in case the size of the constructed partial assignment does not become greater than the width of F at some point. Let $k = \text{width}(F)$ and let $\mathcal{F}_{k-1}$ be the set formed by the initial clauses in F and all the clauses that can be derived from them by resolution using only clauses of width at most $k-1$. Observe that $\mathcal{F}_{k-1}$ is unsatisfiable since it includes the initial clauses of F, but the empty clause does not belong to $\mathcal{F}_{k-1}$. When asked for the value of a variable, Duplicator simply has to answer a value that does not falsify a clause in $\mathcal{F}_{k-1}$. We will show that if the size of the constructed partial assignment α is at most k, then this is always possible. This proves that Duplicator would always score at least $k+1$ points when playing on $\mathcal{F}_{k-1}$, and therefore also at least as many points when playing on F. By the way of contradiction, suppose that Spoiler had a way to falsify a clause in $\mathcal{F}_{k-1}$ always keeping the size of α at most k. Let r be the first round in which Spoiler achieves this. Clearly $r > 0$ since the empty clause does not belong to $\mathcal{F}_{k-1}$. Let α be the partial assignment of size at most $k-1$ constructed just before round r. In round r Spoiler asks for the value of a new variable x. Since for either possible answer of Duplicator one of the clauses in $\mathcal{F}_{k-1}$ is falsified, there must be two clauses $C = C' \cup \{x\}$ and $D = D' \cup \{\overline{x}\}$ in $\mathcal{F}_{k-1}$ and the assignment α falsifies both C' and D'. Since α has size at most $k-1$ and $C' \cup D'$ is falsified by α, this clause has width at most $k-1$. Because this clause can be derived by the resolution of C and D, it belongs to $\mathcal{F}_{k-1}$. But then α would already falsify this clause in round $r-1$ and this contradicts the fact that r was the first round in which a clause in $\mathcal{F}_{k-1}$ was falsified. ■

6.2 A Game Characterization of Tree-like Resolution Space

For the case of tree-like resolution there is also a combinatorial characterization based on a similar 2-person game. This game was introduced by Impagliazzo and Pudlák in [26] and has been used for proving lower bounds on the size of tree-like resolution refutations [26, 6].

The game, which we will just call Game B, is played in rounds on an unsatisfiable formula F in CNF by two players called in this case Prover and Delayer. The players construct in steps a partial assignment for the formula. Prover wants to falsify some initial clause and Delayer tries to retard this as much as possible. In each round Prover chooses a variable in F and asks Delayer for a value for this variable. Delayer can answer either $0, 1$ or $*$. In this last case Prover can choose the truth value (0 or 1) for the variable and

Delayer scores one point. The variable is set to the selected value and the next round begins. When a variable has been assigned a value, it remains with this value until the end of the game. The game ends when a clause in F is falsified by the partial assignment constructed this way. The goal of Delayer is to score as many points as possible and Prover tries to prevent this. The outcome of the game is the number of points scored by Delayer.

Definition 16. *Let F be an unsatisfiable formula in CNF. We denote by $game_B(F)$ the maximum number of points that Delayer can score while playing Game B on F against an optimal strategy for Prover.*

It was shown in [18] that this game provides an exact characterization of the space needed in tree-like resolution.

Theorem 17. *For an unsatisfiable formula F in CNF*

$$\textit{tree-space}(F) = \ \textit{game}_B(F) + 1.$$

Proof. We show first that game$_B(F) + 1$ is an upper bound for the tree-like resolution space.

Let s be the minimum space needed in any tree-like resolution refutation of F. We give a strategy for Delayer for playing the game on F that scores at least $s-1$ points with any strategy of Prover. This is shown by induction on the number of variables in F, n.

For the base case $n = 1$, F contains just one variable and therefore $s \leq 2$. Delayer just needs to answer $*$ to the only variable asked by Prover.

For $n > 1$, let x be the first variable asked by Prover and let $F_{x=1}$ and $F_{x=0}$ the CNF formulas obtained after giving value 1 and 0 respectively to variable x in F. Any tree-like refutation of F requires s pebbles and therefore either

i) the tree-like space for refuting each of $F_{x=1}$ and $F_{x=0}$ is at least $s-1$ or

ii) for one of the formulas the tree-like resolution space is at least s.

Any other possibility would imply that F could be refuted in space less than s.

In the first case Delayer can answer $*$ and scores one point. By induction hypothesis Delayer can score $s-2$ more points playing the game in either of the formulas $F_{x=1}$ or $F_{x=0}$. In the second case Delayer answers the value leading to the formula that requires tree-like resolution space s ($x = 1$ for example) and the game is played on $F_{x=1}$ in the next round.

On the other hand game$_B(F)$ is also a lower bound for the tree-like resolution space. Let us consider a tree-like resolution refutation of F, π, and suppose that Prover and Delayer play the game on F. Delayer follows a strategy scoring at least game$_B(F)$ points and Prover chooses the variables in an order induced by the refutation in the following way: Prover starts at the empty clause in π and in general at the end of a round moves to a clause C. In the next round Prover chooses the resolved variable x from the two parent clauses of C. If Delayer assigns to x a value 0 or 1 then Prover moves to the parent clause that is falsified by the partial assignment and the new round starts. If Delayer assigns x value $*$ then Prover can choose value 0 or 1 for x and moves to the parent clause falsified by the chosen partial assignment. In this case we mark the clause with $*$. The game ends when Prover can move to an initial clause.

For a tree-like refutation π let us denote by $T(\pi)$ the subtree of π formed by all the clauses that can be visited by Prover and the edges joining them in the described game (with a strategy from Delayer scoring at least game$_B(F)$ points). We show that the pebbling number of $T(\pi)$ is at least game$_B(F)+1$ pebbles. Since $T(\pi)$ is a subgraph of π, this implies that tree-like space for F is at least game$_B(F)+1$.

Observe that in any path from the empty clause to an initial clause in $T(\pi)$ there are at least game$_B(F)$ nodes marked with $*$ (branching nodes). Consider any strategy for pebbling the tree $T(\pi)$, and consider the first moment m in which all the paths going from an initial clause to the empty clause contain a pebble. After moment $m-1$ a pebble has to be placed on an initial clause C, and before that, the path going from C to the empty clause is the only path without pebbles. This path contains at least game$_B(F)$ nodes marked with *. In each one of these nodes starts a path going to an initial clause. All these paths are disjoint and they all contain a pebble at instant $m-1$ (otherwise there would be at moment m a path from the empty clause to some initial clause without any pebble). Together with the pebble at moment m, this makes at least game$_B(F)+1$ pebbles. ■

A similar game to Game B, has been defined in [16] and applied for proving lower bounds for the space in resolution and in Res$_k$, a more powerful proof system. In this new game Prover does not ask for a variable like in Game B, but for a clause, and Delayer assigns values to at least one of the variables in the clause.

6.3 Width versus Space

Atserias and Dalmau have used the combinatorial characterization of resolution width in order to prove that this complexity measure bounds the

resolution space from below. The same result had been obtained for the simpler case of tree-like resolution in [17].

Theorem 18. *[2] For an unsatisfiable formula F in CNF*

$$space(F) \geq width(F) - initial\text{-}width(F) + 1.$$

Proof. Let F be an unsatisfiable formula in CNF and w be the width of its largest clause. We show that if width $(F) = k + w$ for some k then space$(F) \geq k$. By the combinatorial characterization of width in Theorem 15 if width$(F) = k + w$, then Spoiler does not have a terminating strategy keeping a partial assignment of size at most $k + w$ when playing Game A on F. Consider a resolution refutation of F using space less than $k + 1$, $F_0, F_1, F_2, \ldots F_m$. We show by induction on i that the set of clauses in each of the formulas F_i is satisfiable, implying that it is not possible to refute F with space less than $k + 1$. Let s_i be the number of clauses in F_i. We give a strategy for Spoiler to construct for each i a partial assignment α_i of size at most s_i satisfying all the clauses in F_i. Since there is no terminating strategy keeping always partial assignments of size bounded by $k+w$, this assignment cannot falsify an initial clause of F.

F_1 is an initial clause in F and therefore satisfiable. α_1 is a partial assignment of size 1 giving value 1 to one of the literals in F_1. This assignment can be obtained by Spoiler by asking the value of all the variables in F_1 until one satisfies the clause, and deleting the values of all the other assigned variables afterwards. For the induction step, if F_i is obtained by adding the resolvent of two clauses to F_{i-1} then clearly α_{i-1} also satisfies F_i since the parent clauses of the resolvent are satisfied by α_{i-1}. If the new formula is obtained by erasing some of the clauses in F_{i-1} then F_i is still satisfiable and a partial assignment α_i of size $\leq s_i$ can be obtained by Spoiler deleting if necessary some of the assigned values of α_{i-1}. The interesting case arises when F_i is obtained by adding some initial clause C to F_{i-1}. In this case Spoiler extends the partial assignment α_i by asking the Duplicator for the values for all the variables in C that are not assigned at this point. There are at most w variables in the clause and therefore the partial assignment has now size at most $s_{i-1} + w \leq k + w$. Since the assignment does not falsify any initial clause, at least one of the literals in C is true in the partial assignment. Spoiler constructs α_i by keeping a restriction of the actual assignment with size at most s_i. This is always possible since at most the assignment of a variable in each clause is needed to satisfy the whole F_i. ■

The width and the space bounds in the resolution of Tseitin formulas for certain expander graphs basically coincide [7, 17]. Since these formulas have initial clauses of constant width, this shows that the space lower bound

in terms of width from Theorem 18 is best possible. As in the case of the relations between width and size mentioned in Section 3, the initial width is also part of the width-space relation. It has to be this way since it is easy to construct formulas with large initial width that need only constant resolution space. For formulas in k-CNF for a constant k this result proves that resolution width lower bounds imply space lower bounds, but this is not necessarily the case for formulas with large initial clauses. A way to avoid this problem is given in [2]. The authors propose there a way to transform a formula F with large initial clauses into an equivalent one F' with initial clauses of constant size and requiring roughly the same resolution width and space as F. This allows the use of the above width-size relation together with width lower bounds to prove space lower bounds for formulas with large initial clauses.

7 Conclusions and Open Problems

Resolution width and space are alternative complexity measures to the well studied resolution size and provide a better understanding of this proof system. Interesting in their own right, these measures have helped to unify many of the existing results about the complexity of resolution. We have reviewed in this survey the tight connections between the different complexity measures. As in other areas in computer science related to memory allocation problems, the pebbling game has proved to be a very useful tool to analyze space requirements in resolution.

Although in the last years many central results relating these complexity measures have been established, several questions are still open. An important one is whether is it possible to obtain a result relating resolution space and size as in Theorem 9 but for the case of general resolution, thus showing that large resolution space implies large resolution size. Trying to prove the opposite, one could search for a family of formulas with polynomial size resolution refutations that need large space. Good candidates for this are the pebbling contradictions of degree 2 (cf. Definition 11). These formulas can be refuted within polynomial size and constant width, but their resolution space requirements are not known. A related problem is whether for every unsatisfiable F $\text{space}(F) \leq \text{width}(F)$.

Concerning the combinatorial characterizations, we have games for the width and the tree-like resolution space. Is there a similar characterization for the space or variable space in general resolution? This would be nice results to complete the picture.

Acknowledgments: I would like to thank Albert Atserias, Eli Ben-Sasson, Juan Luis Esteban, Nicola Galesi and Randall Pruim for very helpful comments and corrections on preliminary versions of this survey.

References

[1] M. Alekhnovich, E. Ben-Sasson, A.A. Razborov, and A. Wigderson. Space complexity in propositional calculus. *SIAM Journal on Computing*, 31(4):1184–1211, 2002.

[2] A. Atserias and V. Dalmau. A combinatorial characterization of resolution width. *18th IEEE Conference on Computational Complexity* 239–247, 2003.

[3] P. Beame and T. Pitassi. Simplified and improved resolution lower bounds. In *37th Annual IEEE Symposium on Foundations of Computer Science* , 274–282, 1996.

[4] E. Ben-Sasson. Size space tradeoffs for resolution. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing* 457–464, 2002.

[5] E. Ben-Sasson and N. Galesi. Space complexity of random formulae in resolution. Random Structures and Algorithms, 23(1) 92–109, 2003.

[6] E. Ben-Sasson, R. Impagliazzo, and A. Wigderson. Near-optimal separation of treelike and general resolution. In *Electronic Colloquium on Computational Complexity, Report TR02-005*, 2000. To appear in Combinatorica.

[7] E. Ben-Sasson and A. Wigderson. Short proofs are narrow – resolution made simple. *Journal of the ACM*, 48(2):149–169, 2001.

[8] M.L. Bonet and N. Galesi. Optimality of Size-Width tradeoffs for Resolution. *Computational Complexity*, 10 (2001), 261-276.

[9] M.L. Bonet, J.L. Esteban, N. Galesi and J. Johannsen. On the relative complexity of resolution refinements and cutting planes proof systems. *SIAM Journal on Computing* 30(5), 1462–1484, 2002.

[10] V. Chvátal and E. Szemerédi. Many hard examples for resolution. *Journal of the ACM* 35, 759–768, 1988.

[11] M. Clegg, J. Edmonds and R. Impagilazzo. Using the Gröbner Basis algorithm to find proofs of unsatisfiability. *Proc. 28th ACM symp. on Theory of Computing*, 239–251, 1996.

[12] S.A. Cook and R.A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic* 44, 36–50, 1979.

[13] S.A. Cook and R. Sethi. Storage requirements for deterministic polynomial time recognizable languages. *Journal of Computer and System Sciences* 13, 25–37, 1976.

[14] M. Davis, G. Logemann and D. Loveland. A machine program for theorem proving. *Communications of the ACM* 5, 394–397, 1962.

[15] J.L. Esteban. Complexity measures for resolution. PhD. Thesis, Dept. L.S.I. Universitat Politècnica de Catalunya, 2003.

[16] J.L. Esteban, N. Galesi and J. Messner. On the complexity of resolution with bounded conjunctions. *Proc. 29th ICALP Colloquium* Springer Lecture Notes in Computer Science 2380, 220–231. Will appear in *Theoretical Computer Science*.

[17] J.L. Esteban and J. Torán. Space bounds for resolution. *Information and Computation*, 171(1):84–97, 2001.

[18] J.L. Esteban and J. Torán. Combinatorial characterization of treelike resolution space. *Information Processing Letters* 87(6), 295–300, 2003.

[19] Z. Galil. On resolution with clauses of bounded size. *SIAM Journal on Computing* 6, 444–459, 1977.

[20] A. Haken. The intractability of resolution. *Theoretical Computer Science*, 39(2-3):297–308, 1985.

[21] H. Kleine-Büning and T. Lettmann. *Aussagenlogik: Deduktion und Algorithmen.* B.G. Teubner, Stuttgart, 1994.

[22] B. Krishnamurthy and R.N. Moll. Examples of hard tautologies in the propositional calculus. Proc. 13th ACM Symposium on Theory of Computation, 28–37, 1981.

[23] Ph.G. Kolaitis and M.Y. Vardi. On the expressive power of Datalog: tools and a case study. *Journal of Computer and System Sciences* 51:110–134, 1995.

[24] T. Lengauer and R.E. Tarjan. Upper and lower bounds on time-space tradeoffs. *Proc. 11th ACM Symposium on Theory of Computing* 262–277, 1979.

[25] N. Pippenger. Pebbling. IBM Technical Report RC 8258, Watson Research Center 1980.

[26] P. Pudlák and R. Impagliazzo. A lower bound for DLL algorithms for k-SAT. *Proc.11th Annual ACM-SIAM Symposium on Discrete Algorithms* 128–136, 2000.

[27] J.A. Robinson. A machine oriented logic based on the resolution principle. *Journal of the ACM* 12(1), 23–41, 1965.

[28] J. Savage. *Models of Computation.* Addison-Wesley, 1998.

[29] J. Torán. Lower bounds for the space used in resolution. In *Proc. 13th Computer Science Logic Conference*, Springer Lecture Notes in Computer Science 1683, 362–373, 1999.

[30] G.S. Tseitin. On the complexity of derivation in propositional calculus. In *Studies in Constructive Mathematics and Mathematical Logic, Part 2.* , pages 115–125. Consultants Bureau, 1968.

[31] A. Urquhart. Hard examples for resolution. *Journal of the ACM* 34, 209–219, 1987

[32] M. Zito An upper bound on the space complexity of random formulae in resolution. In RAIRO - Theoretical Informatics and Applications 36(4), 329–340, 2002.