Annals of Mathematics and Artificial Intelligence
https://doi.org/10.1007/s10472-021-09734-6

Critical Properties and Complexity Measures of Read-Once Boolean Functions

Vadim Lozin[1] · **Mikhail Moshkov**[2]

Accepted: 24 February 2021 Published online: 22 March 2021

Abstract
In this paper, we define a quasi-order on the set of read-once Boolean functions and show that this is a well-quasi-order. This implies that every parameter measuring complexity of the functions can be characterized by a finite set of minimal subclasses of read-once functions, where this parameter is unbounded. We focus on two parameters related to certificate complexity and characterize each of them in the terminology of minimal classes.

Keywords Read-once Boolean function · Certificate complexity · Well-quasi-ordering

Mathematics Subject Classification (2010) 06E30 · 06A07

1 Introduction

A Boolean function f is called *read-once* if it can be represented by a Boolean expression using the operations of conjunction, disjunction and negation, in which every variable appears exactly once. In many applications and theoretical studies of read-once functions, including the present paper, the operation of negation is irrelevant, because the unique occurrence of a negated variable $\overline{x}$ can be renamed as x.

Read-once functions constitute a small but remarkable class of Boolean functions. They enjoy many nice properties and find applications across various fields, such as computational learning theory [2], circuit design [16], game theory [17], fault diagnosis [29], etc. A subclass of read-once functions, known as *linear read-once* [26] or *nested canalyzing* [21] functions, is important in biological applications. The importance of read-once functions is also due to the fact that they are closely related to other mathematical structures, such as

✉ Vadim Lozin
V.Lozin@warwick.ac.uk

Mikhail Moshkov
mikhail.moshkov@kaust.edu.sa

[1] Mathematics Institute, University of Warwick, Coventry, CV4 7AL, UK

[2] CEMSE Division, King Abdullah University of Science and Technology (KAUST), Thuwal, 23955-6900, Saudi Arabia

cographs [20], separable permutations [1], or series-parallel partial orders. For more information on read-once functions we refer the reader to the book [13], which is one of the most comprehensive sources of information on Boolean functions and which devotes a chapter to this class.

All read-once functions have a simple structure, but not all of them are equally simple. With a closer look at this class, we discover a complex world, where many parameters measuring complexity of the functions, such as sensitivity or certificate complexity, can be arbitrarily large. On the other hand, we observe that this world is well-organized. To formalize this notion, we define a quasi-order on the functions in this class and show that this is a well-quasi-order. This implies, in particular, that for every parameter, which is unbounded in the class of read-once functions, there is a collection of "critical points", i.e. minimal subclasses where the parameter jumps to infinity.

The notion of critical classes, also known as critical properties [25], can be viewed as a generalization of the notion of minimal obstructions, which is a customary way of describing downward closed partial orders (lower sets). In the land of permutations, lower sets are known as pattern classes and every pattern class can be described in the terminology of minimal permutations avoiding it. In the world of graphs, there exist different partial orders (minors, topological minors, induced minors, vertex minors, subgraphs, induced subgraphs etc.) and each lower set admits a characterization in terms of minimal obstructions (minimal minors, topological minors, etc.). The universe of Boolean functions also admits different types of orderings (see e.g. [12, 37]) and closed classes of functions admit various characterizations in terms of minimal obstructions (see e.g. [38]). In particular, in [35] the reader can find a characterization of the class of read-once functions in the terminology of minimal subfunctions that do not belong to this class.

To emphasize the importance of the characterization of closed classes of combinatorial structures in terms of minimal obstructions, let us consider the following example. In 1969 Journal of Combinatorial Theory published a paper entitled 'An interval graph is a comparability graph' [19]. One year later, the same journal published another paper entitled 'An interval graph is not a comparability graph' [14]. Each of the two classes is closed under taking induced subgraphs, and for each of them, there is a list of minimal forbidden induced subgraphs. Apparently, in 1969 this list was not available for at least one of them, because forbidden graphs provide a simple way of comparing two classes.

Now let us shift our discussion from lower sets to *families* of lower sets and ask whether a family $\mathcal{A}$ of sets, which is downward closed under the inclusion relation, can be characterized in terms of minimal sets that do not belong to $\mathcal{A}$. In a well-quasi-ordered world, the answer to this question is 'yes'. For instance, the graph minor relation is known to be a well-quasi-order [33], and in the family of minor-closed classes of graphs the planar graphs constitute a unique minimal obstruction for classes of bounded tree-width [32]. The pattern containment relation defined on the set of permutations is generally not a well-quasi-order. However, it may become a well-quasi-order when restricted to specific classes, such as separable permutations. As a result, any property of pattern classes of separable permutations can be described by means of critical classes, i.e. minimal classes that do not possess this property. In particular, in [1] the family of subclasses of separable permutations that have rational generating functions has been characterized by means of two critical classes. Similarly, in [8] the family of subclasses of cographs of bounded linear clique-width has been characterized by means of two subclasses, where this parameter is unbounded.

In the present paper, we develop a similar approach to identifying critical properties of read-once functions. Three most important complexity measures of Boolean functions are sensitivity, block sensitivity and certificate complexity, denoted $s(f)$, $bs(f)$ and $C(f)$,

respectively. It is known that for any Boolean function f we have $s(f) \leq bs(f) \leq C(f)$ and $C(f) \leq bs(f)s(f)$. Also, the solution [18] of a long-standing open problem, known as sensitivity conjecture, shows that $bs(f) \leq s(f)^c$ for a constant c. Therefore, all three parameters are equivalent in any class X of functions in the sense that either all of them are bounded or all are unbounded in X. Moreover, in the universe of read-once functions the three parameters coincide [27] and admit a simple characterization in terms of critical classes: elementary disjunctions, i.e. functions of the form $x_1 \wedge \ldots \wedge x_k$, and elementary conjunctions, i.e. functions of the form $x_1 \vee \ldots \vee x_k$ are the only two unavoidable structures in any read-once function of large certificate complexity. This conclusion has a Ramsey-type flavour and can be derived directly from Ramsey's Theorem by analogy with the fact that cliques (complete subgraphs) and independent sets (edgeless subgraphs) are the only two unavoidable structures in any graph of large order (the number of vertices).

We observe that Ramsey-type arguments is a typical approach to identifying unavoidable structures [36] and many graph parameters have been characterized in this way, see e.g. [11, 24]. With the advent of parameterized complexity, the gallery of graph parameters has been enriched with a myriad of new representatives, such as shrub-depth [15] or distinguishing number [4], that allow to distinguish various new graph properties and provide a fine-grained analysis of their complexity.

To enrich the gallery of Boolean complexity measures, in this paper we introduce two new parameters, both extending certificate complexity, and characterize each of them in terms of critical properties of read-once functions. A graph-theoretic analog of one of these two parameters has been recently shown to be responsible for the Ramsey number to be linear in a class of graphs [6]. In the terminology of critical properties of functions this parameter is characterized by two classes: read-once DNFs (disjunctive normal forms) and read-once CNFs (conjunctive normal forms). Interestingly, translated to the language of separable permutations, these two classes of functions coincide with the two classes of permutations that are critical for classes with rational generating functions.

The organization of the paper is as follows. In Section 2, we introduce basic terminology and notation. In Section 3, we define a quasi-order on the set of read-once functions and show that this is a well-quasi-order. Section 4 is devoted to two new parameters and their characterization in terms of minimal classes of read-once functions, where these parameters are unbounded. Section 5 concludes the paper with a number of remarks and open problems.

2 Read-Once Formulas, Trees, and Functions

Let V be a countable set of variables.

2.1 Formulas

The notion of a *formula over the basis* $\{\vee, \wedge\}$ is defined as follows:

- A variable from V is a formula over the basis $\{\vee, \wedge\}$.
- If F_1 and F_2 are formulas over the basis $\{\vee, \wedge\}$, then the expressions $(F_1 \vee F_2)$ and $(F_1 \wedge F_2)$ are formulas over the basis $\{\vee, \wedge\}$.

Note that in formulas, $\vee$ and $\wedge$ are symbols of Boolean operators. A formula F over the basis $\{\vee, \wedge\}$ is called a *read-once formula* if each variable, that appears in F, appears in F exactly once. We will represent read-once formulas over the basis $\{\vee, \wedge\}$ by read-once trees over the same basis.

2.2 Trees

First, we define the notion of a *binary tree with the output*. This is a directed tree in which we fix a node, called the *output*, and orient all edges towards the output. In this tree, each node either

- has no entering edges, in which case we call it an *input* or
- has exactly two entering edges that are labelled with l and r, in which case we call it a *gate*.

Each node of the tree with the exception of the output has exactly one leaving edge. The output has no leaving edges.

Let v, v_1, v_2, v_3 be nodes of a binary tree with the output. If two edges labelled with l and r leave the nodes v_1 and v_2, respectively, and enter the node v, then we call v_1 and v_2 the *parents of* v. The node v_1 is called the *l-parent of* v and the node v_2 is called the *r-parent of* v. If an edge leaves the node v and enters the node v_3, then we call v_3 the *child of* v.

A *read-once tree over the basis* $\{\vee, \wedge\}$ (ROT) is a binary tree with the output in which inputs are labelled with pairwise different variables from V and each gate is labelled with a symbol of Boolean operator from the set $\{\vee, \wedge\}$. A gate labelled with the symbol $\vee$ is called a *$\vee$-gate*, and a gate labelled with the symbol $\wedge$ is called a *$\wedge$-gate*.

2.3 Correspondence Between Trees and Formulas

There is a one-to-one correspondence between read-once trees and read-once formulas over the basis $\{\vee, \wedge\}$ that can be described as follows.

Let T be a read-once tree. With each node v of T we associate a formula F_v over the basis $\{\vee, \wedge\}$. If v is an input labelled with a variable x then $F_v = x$. Let v be a gate with the l-parent v_1 and the r-parent v_2. If v is a $\vee$-gate, then $F_v = (F_{v_1} \vee F_{v_2})$. If v is a $\wedge$-gate, then $F_v = (F_{v_1} \wedge F_{v_2})$. Let v_o be the output of T. We associate with the read-once tree T the formula $F_T = F_{v_o}$.

Conversely, with each read-once formula F over the basis $\{\vee, \wedge\}$ we associate a ROT T_F in the following way:

- If $F = x$, then T_F contains only one node labelled with the variable x, which is simultaneously the input and the output.
- If $F = (F_1 \vee F_2)$, then T_F consists of ROTs T_{F_1}, T_{F_2}, a $\vee$-gate v, and two edges that leave outputs of T_{F_1} and T_{F_2} and enter v, which is the output of T. These edges are labelled with l and r, respectively.
- If $F = (F_1 \wedge F_2)$, then T_F consists of ROTs T_{F_1}, T_{F_2}, a $\wedge$-gate v, and two edges that leave outputs of T_{F_1} and T_{F_2} and enter v, which is the output of T. These edges are labelled with l and r, respectively.

2.4 Functions

Let F be a formula over the basis $\{\vee, \wedge\}$. We associate with the formula F a Boolean function f_F in the following way ($\vee$ and $\wedge$ are Boolean operators):

- If $F = x$, where x is a variable, then $f_F = x$.
- If $F = (F_1 \vee F_2)$, then $f_F = f_{F_1} \vee f_{F_2}$.
- If $F = (F_1 \wedge F_2)$, then $f_F = f_{F_1} \wedge f_{F_2}$.

Let $x_1, \ldots, x_n$ be all the variables that appear in the formula F. The domain of f_F is the set of valuations of these variables. We will write $f_F = f_F(x_1, \ldots, x_n)$ and say that f_F depends on variables $x_1, \ldots, x_n$.

A Boolean function f is called a *read-once function over the basis* $\{\vee, \wedge\}$ (ROF) if there exists a read-once formula F over the basis $\{\vee, \wedge\}$ such that $f = f_F$. The latter equality means that the functions f and f_F depend on the same variables and have equal values for the same valuations of these variables. If T is a read-once tree, we also denote $f_T = f_{F_T}$.

We say that a variable x_i is *relevant* for a Boolean function $f = f(x_1, \ldots, x_n)$ if there exist two n-tuples $\bar{\alpha}$ and $\bar{\beta}$ in $\{0, 1\}^n$ such that $\bar{\alpha}$ and $\bar{\beta}$ are different only in the i-th digit and $f(\bar{\alpha}) \neq f(\bar{\beta})$.

The following claim can be easily proved by induction on the number of symbols $\vee$ and $\wedge$ in a formula F describing a read-once function $f = f_F$.

Claim 1 *If f is a read-once function depending on variables $x_1, \ldots, x_n$, then each variable x_i is relevant for f.*

Let f be a read-once function depending on variables $x_1, \ldots, x_n$ and F be a read-once formula over the basis $\{\vee, \wedge\}$ such that $f = f_F$. From Claim 1 it follows that $x_1, \ldots, x_n$ are all the variables that appear in the formula F.

3 From Quasi-Ordering to Well-Quasi-Ordering

A binary relation $\leq$ on a set W is a *quasi-order* (also known as *preorder*) if it is reflexive and transitive. Two elements $x, y \in W$ are said to be *comparable* with respect to $\leq$ if either $x \leq y$ or $y \leq x$. Otherwise, x and y are *incomparable*. A set of pairwise comparable elements is called a *chain* and a set of pairwise incomparable elements an *antichain*. If $x \leq y$ and $y \not\leq x$, we write $x < y$. A chain $x_1 > x_2 > \ldots$ is called *strictly decreasing*. A quasi-order $(W, \leq)$ is a *well-quasi-order* if it contains neither infinite strictly decreasing chains nor infinite antichains.

In order to define a quasi-order on the set of read-once functions, we start by defining the operation of removal of variables.

3.1 Removal of Variables from Read-Once Functions

The operation of removal of a variable is applicable only to functions depending on at least 2 variables. To define this operation, we consider a read-once function f and a variable x on which f depends.

Assume first that f is given together with a read-once formula F representing it. Speaking informally, the operation of removal of x from f can be viewed as the operation of "erasing" x from F and then modifying F accordingly, to make it a read-once formula again. In the terminology of read-once trees, this modification can be formally described as follows.

Let $T = T_F$ and let v_x be the input of T labelled by x. The removal of v_x from T is the operation that transforms T into another read-once tree, denoted $T - v_x$, in the following way. We denote by v_1 the child of v_x, by v_2 the second parent of v_1, and by v_3 the child of v_1, if v_1 is not the output. The removal of v_x consists in removing the nodes v_x and v_1, and the edges leaving the nodes v_x, v_2, and v_1. If v_1 is not the output in T, then we also add

a new edge from v_2 to v_3 (see Fig. 1 for an illustration). If v_1 is the output in T, then v_2 becomes the output in $T - v_x$.

This operation also is equivalent to substituting x with an appropriate constant, 0 or 1, in F.

- If v_1 is a $\vee$-gate, then the removal of v_x from T is equivalent to the substitution of x with 0, in which case the sub-formula $(x \vee F_{v_2})$ of F transforms into the sub-formula F_{v_2}. According to Claim 1, all the $n - 1$ variables of the function obtained by this substitution are relevant. Note that if we substitute x with 1, then all the variables from the sub-formula F_{v_2} become irrelevant and the function obtained by this substitution has strictly fewer than $n - 1$ relevant variables.
- If v_1 is a $\wedge$-gate, then the removal of v_x is equivalent to the substitution of the variable x with 1, in which case the sub-formula $(x \wedge F_{v_2})$ transforms into F_{v_2} and all the $n - 1$ variables of the function obtained by this substitution remain relevant. On the other hand, if we substitute x with 0, then all the variables from the sub-formula F_{v_2} become irrelevant, i.e. the function obtained by this substitution has strictly fewer than $n - 1$ relevant variables.

Now we give a definition of the operation of removal of a variable from a read-once function f, which is irrespective of the formula representing f. Let $f = f(x_1, \ldots, x_n)$. For a variable x_i, *exactly one* of the two functions

$$f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \text{ and } f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$$

depends on $n-1$ variables. We denote this function by $f - x_i$ and say that $f - x_i$ is obtained from f by the removal of variable x_i.

Remark 1 It is not difficult to see that $f - x_i = f_{T - v_{x_i}}$ for *any* tree T representing f.

3.2 Quasi-Ordering and Closed Classes of ROTs and ROFs

In this section, we define quasi-orders on the sets of read-once trees and read-once functions. To this end, we need to introduce two more operations.

The operation *renaming of variables* applies to the variables attached to inputs and uses only the variables from the set V. After renaming, all variables must be pairwise different.

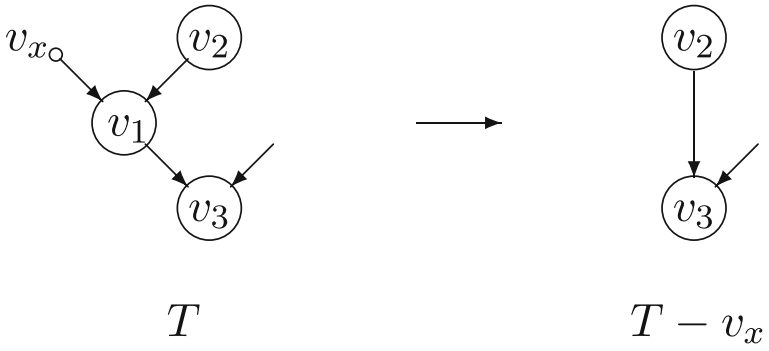

Fig. 1 Removal of an input from a read-once tree

The operation *renaming of labels* applies to the labels l and r attached to edges, and after renaming any two edges entering the same node must have different labels, i.e. one edge must be labelled with l and the other one with r.

Definition 1 We will say that a read-once tree T_1 is a *subtree* of a read-once tree T_2 if T_1 can be obtained from T_2 by a (possibly empty) sequence of three operations: removal of inputs, renaming of variables and renaming of labels.

In a similar way, we define the notion of a subfunction. The only difference is that the operation of renaming of labels is irrelevant in this case, since this operation applied to a read-once tree T does not change the function corresponding to T.

Definition 2 We will say that a read-once function f_1 is a *subfunction* of a read-once function f_2 if f_1 can be obtained from f_2 by a (possibly empty) sequence of two operations: removal of variables and renaming of variables.

The notions of subtree and subfunction define binary relations of the sets of trees and functions, respectively. It is not difficult to see that these relations are reflexive and transitive, i.e. they are quasi-orders. However, they are not partial orders, since they are not antisymmetric. Indeed, the following two functions

$$f_1(x_1, x_2, x_3) = x_1 \wedge (x_2 \vee x_3) \text{ and } f_2(x_1, x_2, x_3) = x_2 \wedge (x_1 \vee x_3)$$

are subfunctions of each other, but $f_1 \not\equiv f_2$, because $f_1(0, 1, 1) \neq f_2(0, 1, 1)$.

The notions of subtree and subfunction naturally lead to the notions of (downward) closed classes of trees and functions, respectively.

Definition 3 We will say that a set $\mathcal{T}$ of read-once trees is a *closed class* if $T \in \mathcal{T}$ implies $T' \in \mathcal{T}$ for every subtree T' of T.

Definition 4 We will say that a set $\mathcal{F}$ of read-once functions is a *closed class* if $f \in \mathcal{F}$ implies $f' \in \mathcal{F}$ for every subfunction f' of f.

Remark 2 When we talk about *closed* classes of trees, the actual labelling of the input nodes and the edges of a tree is irrelevant, since all possible valid labellings of the tree are present in the class. This allows us to consider *partially labelled* read-once trees, i.e. trees in which the input nodes and the edges are not labelled. Each partially labelled ROT T describes a set of ROTs obtained from T by labelling the inputs with pairwise different variables from V and by labelling the edges so that any two edges entering the same node are labelled one with l and the other one with r.

Let X be a set of read-once functions (trees). We denote by $[X]$ the closure of X, i.e. the set consisting of all functions (trees) in X and all their subfunctions (subtrees). If κ is a numerical parameter defined on the functions (trees) in X, then we call X κ-bounded if there is a constant c such that $\kappa(f) \leq c$ for every function (tree) f in X.

3.3 Well-Quasi-Ordering of Read-Once Functions

In this section, we show that the quasi-order defined in Section 3.2 is a well-quasi-oder. To this end, we use the notion of quasi-embedding that can be defined as follows. A map

$f : (X, \leq) \to (Y, \preceq)$ is called a *quasi-embedding* if, for all $a, b \in X$, $f(a) \preceq f(b) \implies a \leq b$. It is known (and not difficult to see) that, if there exists a quasi-embedding $X \to Y$, and $(Y, \preceq)$ is a well-quasi-order, then $(X, \leq)$ is a well-quasi-order too.

By mapping a read-once function to any read-once tree representing it we obtain a quasi-embedding, which is not difficult to see. Therefore, it suffices to show that the set of read-once trees is well-quasi-ordered under the subtree relation. This will be done by defining a quasi-embedding of this set into the set of all rooted trees well-quasi-ordered by the topological minor relation.

To make the reduction easier, we assume, without loss of generality, that the edges of a rooted tree are directed and that all of them are directed towards the root. A *subdivision of an edge* $x \to y$ by a new vertex z is the operation of replacement of $x \to y$ with two new edges $x \to z$ and $z \to y$. A tree T is a *subdivision of a tree* T' if T can be obtained from T' by a sequence of edge subdivisions. A rooted tree T_1 is a *topological minor* of a rooted tree T_2 if there is a subdivision T' of T_1 that can be obtained from T_2 by deleting some nodes.

If T_1 is a topological minor of T_2, then there is an injective mapping ϕ of the node set of T_1 into the node set of T_2 such that for every edge $x \to y$ in T_1 there is a directed path from $\phi(x)$ to $\phi(y)$ in T_2. Without loss of generality, we assume that if y is a child of x_1 and x_2 in T_1, then $\phi(y)$ is the unique common node of the two directed paths connecting $\phi(x_1)$ and $\phi(x_2)$ to $\phi(y)$ in T_2. We also assume that the nodes of the trees are labelled by elements from a well-quasi-ordered set $(Q, \leq)$ and require that ϕ maps a node x of T_1 with label $l(x) \in Q$ to a node y of T_2 with label $l(y) \in Q$ such that $l(x) \leq l(y)$. We will say that a mapping ϕ satisfying this property *respects the labels*. In this terminology the famous Kruskal's theorem for trees [22] can be stated as follows.

Theorem 1 *The set of all rooted trees, whose nodes are equipped with labels from a well-quasi-ordered set, is well-quasi-ordered under the topological minor relation that respects the labels.*

We now use this result in order to show that read-once trees are well-quasi-ordered under the subtree relation. To define a labelling of the node set of a read-once tree T we use symbols $\vee$ and $\wedge$ for the gate nodes and symbol $\emptyset$ for the input nodes, and assume that $\emptyset \leq \vee$ and $\emptyset \leq \wedge$, while $\vee$ and $\wedge$ are incomparable.

Lemma 1 *Let T_1 and T_2 be two read-once trees, whose nodes are equipped with labels from $\{\vee, \wedge, \emptyset\}$. If T_1 is a topological minor of T_2 with a mapping ϕ that respects the labels, then T_1 is a subtree of T_2.*

Proof Without loss of generality we may assume that ϕ maps an input x of T_1 to an input $\phi(x)$ of T_2, because if it is not the case, ϕ can be easily transformed into a mapping satisfying this property by replacing $\phi(x)$ with any input of T_2 for which there is a directed path to $\phi(x)$. Then the desired conclusion can be derived by induction on the number k of inputs of T_2 that do not belong to the set of images of ϕ. If $k = 0$, then there is a bijection between the node set of T_1 and the node set of T_2, because both trees are binary. Therefore, since T_1 is a topological minor of T_2, the two trees coincide.

Assume now that $k > 0$ and let x be an input of T_2, which is not an image of ϕ. The removal of x transforms T_2 into another read-once tree T_2', which is a topological minor of T_2. Indeed, the inverse operation can be viewed as a subdivision of an edge and addition of

a node (examine Fig. 1 from right to left for an illustration). It is also not difficult to see that T_2' contains T_1 as a topological minor. Indeed, the child of x that disappears together with x cannot be an image of any node of T_1, because T_1 is a binary tree and the inputs of T_1 are mapped to inputs of T_2. By induction assumption, T_1 is a subtree of T_2' and hence T_1 is a subtree of T_2. □

This lemma proves more than just well-quasi-orderability of read-once trees, because Kruskal's tree theorem admits various generalizations. In particular, Nash-Williams proved in [31] that infinite (in addition to finite) trees are better-quasi-ordered, and this was later strengthened by Laver (in [23], Theorem 2.2) to labelled infinite trees, provided the set of labels is better-quasi-ordered (which is the case for any finite set).

The full definition of better-quasi-ordering is technical, and outside of the scope of this paper (see, e.g., [3] for a short introduction). What is important to us is that a quasi-embedding $X \to Y$ into a better-quasi-order $(Y, \preceq)$ implies that $(X, \leq)$ is a better-quasi-order too (see, e.g., [3], Lemma 5.3). Thus Lemma 1 shows that the set of read-once trees is better-quasi-ordered.

The additional strength of better-quasi-ordering can be summarised with the following proposition (see, e.g., [3]), where $(X, \leq)$ is an arbitrary quasi-order and $\mathcal{L}(X)$ is the set of downward closed sets of X.

Proposition 1 *If $(X, \leq)$ is a better-quasi-order, then $(X, \leq)$ is a well-quasi-order and $(\mathcal{L}(X), \subseteq)$ is a better-quasi-order.*

The above discussion leads us to the following conclusion.

Corollary 1 *The set of read-once functions is well-quasi-ordered under the subfunction relation and the set of closed classes of read-once functions is well-quasi-ordered under the inclusion relation.*

One important consequence of well-quasi-orderability of the set of closed classes under inclusion can be stated as follows.

Corollary 2 *For every parameter κ, which is unbounded on the set of read-once functions, there is a finite collection of minimal closed subclasses of read-once functions, where this parameter is unbounded.*

This result suggests a quasi-order on the set of parameters and a way of comparing two parameters. For two parameters κ_1 and κ_2, we will write $\kappa_1 \succeq \kappa_2$ if the family of κ_1-bounded closed classes contains the family of κ_2-bounded closed classes.

Corollary 3 *If κ_1 and κ_2 are two parameters, then $\kappa_1 \succeq \kappa_2$ if and only if for every minimal closed class $\mathcal{F}_1$, where κ_1 is unbounded, there is a minimal closed class $\mathcal{F}_2$, where κ_2 is unbounded, such that $\mathcal{F}_2 \subseteq \mathcal{F}_1$.*

Proof Assume first that for every minimal closed class $\mathcal{F}_1$ where κ_1 is unbounded, there is a minimal closed class $\mathcal{F}_2$ where κ_2 is unbounded such that $\mathcal{F}_2 \subseteq \mathcal{F}_1$. Let $\mathcal{X}$ be a κ_2-bounded closed class and suppose that κ_1 is unbounded in this class. Then $\mathcal{X}$ contains a minimal class $\mathcal{F}_1$, where κ_1 is unbounded, which in turn contains a minimal class $\mathcal{F}_2$, where

κ_2 is unbounded. This contradicts our assumption that $\mathcal{X}$ is κ_2-bounded and proves that $\mathcal{X}$ is κ_1-bounded, i.e. $\kappa_1 \succeq \kappa_2$.

Conversely, suppose $\kappa_1 \succeq \kappa_2$, and let $\mathcal{F}_1$ be a minimal closed class where κ_1 is unbounded. Since $\kappa_1 \succeq \kappa_2$, κ_2 is also unbounded in $\mathcal{F}_1$, and by Corollary 2, $\mathcal{F}_1$ contains a minimal closed class $\mathcal{F}_2$ where κ_2 is unbounded, as required. □

Finally, we observe that the quasi-order defined for parameters in the universe of read-once functions is, in fact, a well-quasi-order due to better-quasi-orderability of the functions.

4 Parameters

As we mentioned in the introduction, three important complexity measures of Boolean functions are certificate complexity, block sensitivity, and sensitivity. For read-once functions, these three parameters coincide, as was shown in [27], and therefore, we define only one of them.

Definition 5 Let $f(x_1, \ldots, x_n)$ be a non-constant Boolean function. The *certificate complexity* $C(f, \bar{\delta})$ *of the function* f *on the tuple* $\bar{\delta} = (\delta_1, \ldots, \delta_n) \in \{0, 1\}^n$ is the minimum natural number m such that there exist indices $i_1, \ldots, i_m \in \{1, \ldots, n\}$ for which the function f is constant on the set of tuples $\{(\sigma_1, \ldots, \sigma_n) : (\sigma_1, \ldots, \sigma_n) \in \{0, 1\}^n, \sigma_{i_1} = \delta_{i_1}, \ldots, \sigma_{i_m} = \delta_{i_m}\}$. The value $C_0(f) = \max\{C(f, \bar{\delta}) : \bar{\delta} \in \{0, 1\}^n, f(\bar{\delta}) = 0\}$ is called 0-*certificate complexity* of f. The value $C_1(f) = \max\{C(f, \bar{\delta}) : \bar{\delta} \in \{0, 1\}^n, f(\bar{\delta}) = 1\}$ is called 1-*certificate complexity* of f. The *certificate complexity* of f is $C(f) = \max\{C_0(f), C_1(f)\}$.

Informally, the certificate complexity of a Boolean function f defined on an n-dimensional hypercube can be described as follows. On a Boolean point $\bar{\delta}$ the certificate complexity equals n minus the dimension of a largest sub-hypercube containing $\bar{\delta}$ that defines a constant function. Then $C_0(f)$ is the maximum of this measure taken over 0 points, $C_1(f)$ is the maximum taken over 1 points, and $C(f)$ is the total maximum taken over all points of the hypercube.

It is not difficult to see (in particular, it follows from Lemma 2 below) that in the class of read-once functions certificate complexity can be arbitrarily large. By Lemma 2, this parameter is unbounded even in the class $\mathcal{F}_\wedge$ of elementary conjunctions, i.e. functions of the form $x_1 \wedge \ldots \wedge x_n$, and in the class $\mathcal{F}_\vee$ of elementary disjunctions, i.e. functions of the form $x_1 \vee \ldots \vee x_n$. On the other hand, any closed class of read-once functions excluding at least one elementary conjunction and at least one elementary disjunction is finite, i.e. consists of finitely many functions up to renaming of variables, which is not difficult to see and which also follows from Lemmas 3 and 4 below. Therefore, the characterization of certificate complexity in the terminology of minimal classes is trivial: certificate complexity is bounded in a closed class $\mathcal{F}$ of read-once functions if and only if $\mathcal{F}$ contains neither $\mathcal{F}_\wedge$ nor $\mathcal{F}_\vee$.

We observe that one more important parameter, known as deterministic decision tree complexity, admits the same characterization in terms of minimal classes, because it is bounded below by block sensitivity and above by the square of certificate complexity [9]. Therefore, deterministic decision tree complexity is bounded in a closed class $\mathcal{F}$ if and only if $\mathcal{F}$ is finite up to renaming of variables in functions, similarly to certificate complexity, block sensitivity, and sensitivity. In other words, all these four parameters are equivalent

with respect to the quasi-order defined in the previous section. Moreover, all of them lie at the bottom of the hierarchy that describes this quasi-order in the sense that there are no parameters that lie strictly below them.

To extend this hierarchy, we introduce two new complexity measures of a Boolean function $f(x_1, \ldots, x_n)$:

$$\alpha(f) = \min\{C_0(f), C_1(f)\} \text{ and } \beta(f) = n/C(f).$$

These parameters are closely related to the complexity of nondeterministic decision trees computing f. In deterministic decision trees, each internal node is labelled with a variable and has two leaving edges labelled with 0 and 1, respectively. In nondeterministic decision trees, the root is not labelled with a variable. In this node, we choose nondeterministically the first query (variable). All other internal nodes are labelled with variables, but each such node can have more than one leaving edge labelled with 0 and more than one leaving edge labelled with 1. We distinguish conventional nondeterministic decision trees that accept all tuples from $\{0, 1\}^n$, 0-nondeterministic decision trees that accept only tuples $\bar{\delta} \in \{0, 1\}^n$ for which $f(\bar{\delta}) = 0$, and 1-nondeterministic decision trees that accept only tuples $\bar{\delta} \in \{0, 1\}^n$ for which $f(\bar{\delta}) = 1$. Then $C(f)$ is the minimum depth of a conventional nondeterministic decision tree computing f, $C_0(f)$ is the minimum depth of a 0-nondeterministic decision tree computing f, and $C_1(f)$ is the minimum depth of a 1-nondeterministic decision tree computing f. Note that 1-nondeterministic decision trees were studied in [28, 30] as decision trees computing Boolean functions strongly nondeterministically.

If $\mathcal{F}$ is an α-bounded class of read-once functions, then for each function in $\mathcal{F}$, there is either a 0-nondeterministic or 1-nondeterministic decision tree of bounded depth. If $\mathcal{F}$ is a β-bounded class, then there exists a positive constant t such that, for each function $f(x_1, \ldots, x_n)$ in $\mathcal{F}$, $n/C(f) \leq t$ and $C(f) \geq n/t$, i.e., the minimum depth of a conventional nondeterministic decision tree computing f is at least n/t.

By definition, α is upper bounded by certificate complexity. It is also interesting to observe that, for all read-once functions f, $\beta(f) \leq \alpha(f)$. To see this, consider a read-once function f on n inputs. Without loss of generality, suppose that $C_0(f) \leq C_1(f)$. Then, by definition, $C(f) = C_1(f)$, $\alpha(f) = C_0(f)$, and $\beta(f) = n/C(f) = n/C_1(f)$. By Lemma 7, we have $n \leq C_0(f)C_1(f)$, and dividing through by $C_1(f)$, we conclude that $\beta(f) = n/C_1(f) \leq C_0(f) = \alpha(f)$.

From the double inequality $\beta(f) \leq \alpha(f) \leq C(f)$ we conclude that the family of C-bounded classes is contained in the family of α-bounded classes, which in turn is contained in the family of β-bounded classes. In fact, both containments are *proper*, i.e. neither C can be upper bounded by a function of α, nor α can be upper bounded by a function of β, which follows, in particular, from our characterization of these parameters in terms of critical classes. We start with some auxiliary results.

4.1 Auxiliary results

Let T be a read-once tree. We denote $C_0(T) = C_0(f_T)$, $C_1(T) = C_1(f_T)$. It is clear that, for each input v of the ROT T, $C_0(f_v) = C_1(f_v) = 1$. We can find values of $C_0(f_v)$ and $C_1(f_v)$ for each gate v of T using the following statement.

Lemma 2 *([27], Lemma 2) Let T be a ROT, v be a gate of T, and v_1, v_2 be parents of v.*

If v is a $\vee$-gate, then

$$C_0(f_v) = C_0(f_{v_1}) + C_0(f_{v_2}),$$
$$C_1(f_v) = \max\{C_1(f_{v_1}), C_1(f_{v_2})\}.$$

If v is a $\wedge$-gate, then

$$C_0(f_v) = \max\{C_0(f_{v_1}), C_0(f_{v_2})\},$$
$$C_1(f_v) = C_1(f_{v_1}) + C_1(f_{v_2}).$$

We denote by $h_\vee(T)$ the *$\vee$-depth of the ROT T* which is the maximum number of $\vee$-gates in a path from an input to the output of T. We denote by $h_\wedge(T)$ the *$\wedge$-depth of the ROT T* which is the maximum number of $\wedge$-gates in a path from an input to the output of T. The following statement gives us lower and upper bounds on the value $C_0(T)$ depending on the $\vee$-depth $h_\vee(T)$ of T.

Lemma 3 *Let T be a ROT. Then $h_\vee(T) + 1 \leq C_0(T) \leq 2^{h_\vee(T)}$.*

Proof We prove by induction on $h_\vee(T)$ that $C_0(T) \leq 2^{h_\vee(T)}$. Let $h_\vee(T) = 0$. Then T contains only $\wedge$-gates. Using Lemma 2 and the fact that $C_0(f_v) = 1$ for any input v of T we obtain $C_0(T) = 1$. Therefore the considered inequality holds if $h_\vee(T) = 0$.

Let $n \geq 0$ and $C_0(T') \leq 2^{h_\vee(T')}$ for any ROT T' with $h_\vee(T') \leq n$. Consider a ROT T with $h_\vee(T) = n + 1$. This ROT contains at least one $\vee$-gate. Using Lemma 2 one can show that $C_0(T) \geq 2$. Let v be a gate of T for which $C_0(f_v) = C_0(T)$ and the length of path from v to the output of T is maximum. Let us show that v is a $\vee$-gate. Assume the contrary. By Lemma 2, $C_0(f_v) = C_0(f_{v'})$ for a parent v' of v, but this is impossible since v' is a gate and the length of path from v' to the output is greater than the length of path from v to the output. Therefore v is a $\vee$-gate. Let v_1 and v_2 be parents of v, and T_{v_1} and T_{v_2} be subROTs of T with outputs v_1 and v_2, respectively. It is clear that $h_\vee(T_{v_1}) \leq n$ and $h_\vee(T_{v_2}) \leq n$. Using the inductive hypothesis we obtain that $C_0(T_{v_1}) \leq 2^n$ and $C_0(T_{v_2}) \leq 2^n$. By Lemma 2, $C_0(f_v) \leq 2^{n+1}$. Therefore $C_0(T) \leq 2^{h_\vee(T)}$.

Let us show that $h_\vee(T) + 1 \leq C_0(T)$. To this end, we consider a path from an input of T to the output which contains $m = h_\vee(T)$ $\vee$-gates $v_1, \ldots, v_m$. Using Lemma 2 and the fact that $C_0(f_v) = 1$ for any input v of T one can show that $C_0(f_{v_i}) \geq i + 1$ for $i = 1, \ldots, m$ and $C_0(T) \geq C_0(f_{v_m})$. □

Similarly, we can prove the following statement which gives us lower and upper bounds on the value $C_1(T)$ depending on the $\wedge$-depth $h_\wedge(T)$ of T.

Lemma 4 *Let T be a ROT. Then $h_\wedge(T) + 1 \leq C_1(T) \leq 2^{h_\wedge(T)}$.*

4.2 Parameter α

In this section we study the parameter

$$\alpha(f) = \min\{C_0(f), C_1(f)\}$$

and characterize the family of α-bounded closed classes of read-once functions in terms of minimal closed classes, where this parameter is unbounded.

We start by characterizing α-bounded closed classes of read-once trees in Section 4.2.1 and then extend our results to α-bounded closed classes of read-once functions in Section 4.2.2.

4.2.1 α-Bounded Closed Classes of Read-Once Trees

For a read-once tree T, we define $\alpha(T) = \alpha(f_T)$. The following result provides a criterion for a set of read-once trees to be α-bounded.

Lemma 5 *A set $\mathcal{T}$ of ROTs is α-bounded if and only if it can be represented in the form $\mathcal{T} = \mathcal{T}_\vee \cup \mathcal{T}_\wedge$ where $\mathcal{T}_\vee$ is $h_\vee$-bounded and $\mathcal{T}_\wedge$ is $h_\wedge$-bounded.*

Proof Let $\mathcal{T}$ be α-bounded and let a be a natural number such that $\alpha(T) \le a$ for each tree $T \in \mathcal{T}$. Denote $\mathcal{T}_\vee = \{T : T \in \mathcal{T}, C_0(T) \le a\}$ and $\mathcal{T}_\wedge = \{T : T \in \mathcal{T}, C_1(T) \le a\}$. It is clear that $\mathcal{T} = \mathcal{T}_\vee \cup \mathcal{T}_\wedge$. From Lemmas 3 and 4 it follows that $h_\vee(T) \le a-1$ for any $T \in \mathcal{T}_\vee$ and $h_\wedge(T) \le a-1$ for any $T \in \mathcal{T}_\wedge$. Therefore $\mathcal{T}_\vee$ is $h_\vee$-bounded and $\mathcal{T}_\wedge$ is $h_\wedge$-bounded.

Let $\mathcal{T}$ can be represented in the form $\mathcal{T} = \mathcal{T}_\vee \cup \mathcal{T}_\wedge$ where $\mathcal{T}_\vee$ is $h_\vee$-bounded and $\mathcal{T}_\wedge$ is $h_\wedge$-bounded. Then there exists a natural number b such that $h_\vee(T) \le b$ for any ROT $T \in \mathcal{T}_\vee$, and there exists a natural number c such that $h_\wedge(T) \le c$ for any ROT $T \in \mathcal{T}_\wedge$. From Lemmas 3 and 4 it follows that $C_0(T) \le 2^b$ for any $T \in \mathcal{T}_\vee$ and $C_1(T) \le 2^c$ for any $T \in \mathcal{T}_\wedge$. Denote $a = \max\{2^b, 2^c\}$. Then $\alpha(T) \le a$ for any ROT $T \in \mathcal{T}$. □

According to Remark 2, when we deal with *closed* classes of read-once trees, the actual labelling of the input nodes and the edges of a tree is irrelevant, allowing us to consider partially labelled read-once trees. For natural k and m, let us denote by $\Phi_1(k, m)$, $\Phi_2(k, m)$, $\Phi_3(k, m)$, $\Phi_4(k, m)$ partially labelled read-once trees shown in Fig. 2. Note that each of the partially labelled ROTs $\Phi_1(0, 0)$ and $\Phi_2(0, 0)$ consists of two inputs and one gate. Each of the partially labelled ROTs $\Phi_3(0, 0)$, $\Phi_4(0, 0)$ consists of one input.

Denote

$$\mathcal{T}_1 = [\{\Phi_1(k, m) \ : \ k, m = 0, 1, 2, \ldots\}],$$
$$\mathcal{T}_2 = [\{\Phi_2(k, m) \ : \ k, m = 0, 1, 2, \ldots\}],$$
$$\mathcal{T}_3 = [\{\Phi_3(k, m) \ : \ k, m = 0, 1, 2, \ldots\}],$$
$$\mathcal{T}_4 = [\{\Phi_4(k, m) \ : \ k, m = 0, 1, 2, \ldots\}].$$

Using Lemma 5 we obtain that the classes $\mathcal{T}_1, \ldots, \mathcal{T}_4$ are not α-bounded.

Lemma 6 *Let $\mathcal{T}$ be a closed class of ROTs which is not α-bounded. Then there exists $i_0 \in \{1, \ldots, 4\}$ such that $\mathcal{T}_{i_0} \subseteq \mathcal{T}$.*

Proof Using Lemma 5 we obtain that, for each natural t, there exists a ROT $T_t \in \mathcal{T}$ such that $h_\vee(T_t) \ge t$ and $h_\wedge(T_t) \ge t$. In the ROT T_t, there exist two paths $\pi_t^\vee$ and $\pi_t^\wedge$ each from an input to the output such that $\pi_t^\vee$ contains at least t $\vee$-gates and $\pi_t^\wedge$ contains at least t $\wedge$-gates. Denote by v the first common node of $\pi_t^\vee$ and $\pi_t^\wedge$. Let $\pi_t^\vee$ contain a_t $\vee$-gates before v and b_t $\vee$-gates beginning with v, and let $\pi_t^\wedge$ contain c_t $\wedge$-gates before v and d_t $\wedge$-gates beginning with v. Then $a_t + b_t \ge t$ and $c_t + d_t \ge t$. We denote by τ_t the path in T_t from the node v to the output of T_t, i.e. the common part of the paths $\pi_t^\vee$ and $\pi_t^\wedge$.

We now prove that, for each natural k, the class $\mathcal{T}$ contains a ROT $\Phi_i(k, k)$ for some $i \in \{1, \ldots, 4\}$. To this end, we consider a number of cases.

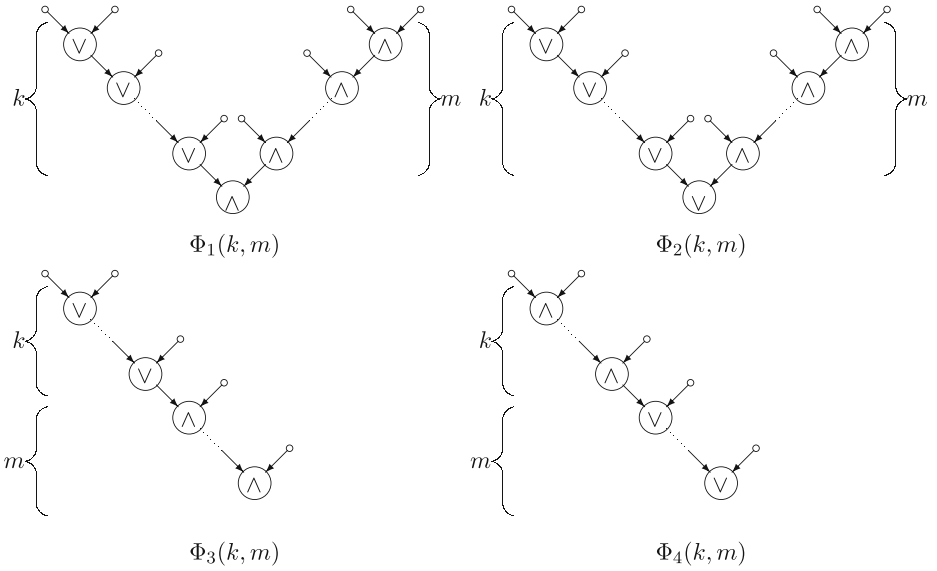

Fig. 2 Partially labelled ROTs $\Phi_1(k,m)$, $\Phi_2(k,m)$, $\Phi_3(k,m)$, $\Phi_4(k,m)$

Case 1. Let the sequences $a_1, a_2, \ldots$ be not bounded from above. Then there exists an infinite subsequence of this sequence $a_{j_1}, a_{j_2}, \ldots$ such that $a_{j_1} < a_{j_2} < \ldots$. We now consider two subcases of this case.

Case 1a. Let the sequence $c_{j_1}, c_{j_2}, \ldots$ be not bounded from above. Then there exists a natural p such that $a_{j_p} \geq k$ and $c_{j_p} \geq k$. By the removal of inputs, we can transform the ROT T_{j_p} into a ROT $\Phi_i(k,k)$ with $i \in \{1,2\}$. This ROT belongs to $\mathcal{T}$.

Case 1b. Let the sequence $c_{j_1}, c_{j_2}, \ldots$ be bounded from above. In this case, the sequence $d_{j_1}, d_{j_2}, \ldots$ is not bounded from above. Therefore there exists a natural p such that $a_{j_p} \geq k$ and $d_{j_p} \geq k$. By the removal of inputs, we can transform the ROT T_{j_p} into a ROT $\Phi_3(k,k)$. This ROT belongs to $\mathcal{T}$.

Case 2. The case when the sequence $c_1, c_2, \ldots$ is not bounded from above can be analyzed by analogy with Case 1.

Case 3. Let both sequences $a_1, a_2, \ldots$ and $c_1, c_2, \ldots$ be bounded from above. Then there exists a natural m such that $b_t \geq t - m$ and $d_t \geq t - m$ for $t = 1, 2, \ldots$. Let p be a natural number for which $b_p \geq 2k$ and $d_p \geq 2k$, and $\tau_p = v_1, \ldots, v_l$ be the common part of the paths $\pi_p^\vee$ and $\pi_p^\wedge$ in the ROT T_p. Let j be the minimum number from the set $\{1, \ldots, l\}$ such that the path $v_1, \ldots, v_j$ contains k $\vee$-gates or k $\wedge$-gates. Assume without loss of generality that the path $v_1, \ldots, v_j$ contains k $\vee$-gates. Then this path contains at most $k-1$ $\wedge$-gates. Therefore the path $v_{j+1}, \ldots, v_l$ contains at least $k+1$ $\wedge$-gates. By the removal of inputs, we can transform the ROT T_p into a ROT $\Phi_i(k,k)$ with $i \in \{3,4\}$. This ROT belongs to $\mathcal{T}$.

From the above analysis. we now know that, for each natural k, there exists $i \in \{1, \ldots, 4\}$ such that $\Phi_i(k,k) \in \mathcal{T}$. Therefore, there exists $i_0 \in \{1, \ldots, 4\}$ and an infinite sequence of natural numbers $k_1, k_2, \ldots$ such that $k_1 < k_2 < \ldots$ and $\Phi_{i_0}(k_j, k_j) \in \mathcal{T}$ for $j = 1, 2, \ldots$. Since $\mathcal{T}$ is closed under removal of inputs, we obtain $\mathcal{T}_{i_0} \subseteq \mathcal{T}$. □

Theorem 2 *Let $\mathcal{T}$ be a closed class of ROTs. Then $\mathcal{T}$ is α-bounded if and only if $\mathcal{T}$ contains none of the classes $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4$.*

Proof Let $\mathcal{T}$ be α-bounded. Then $\mathcal{T}$ contains none of the classes $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4$ since, according to Lemma 5, the classes $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4$ are not α-bounded.

Let $\mathcal{T}$ contain none of the classes $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4$. Then $\mathcal{T}$ is α-bounded, because otherwise, by Lemma 6, $\mathcal{T}$ contains at least one class from the set $\{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4\}$. □

4.2.2 α-Bounded Closed Classes of Read-Once Functions

For all $k, m = 0, 1, 2, \ldots$ such that $k + m \geq 1$, we define the following two read-once functions:

$$\varphi_1(k, m) = (x_1 \vee \ldots \vee x_k) \wedge y_1 \wedge \ldots \wedge y_m,$$
$$\varphi_2(k, m) = x_1 \vee \ldots \vee x_k \vee (y_1 \wedge \ldots \wedge y_m).$$

Also, for $i \in \{1, 2\}$, we denote $\mathcal{F}_i = [\{\varphi_i(k, m) : k, m = 0, 1, 2, \ldots, k + m \geq 1\}]$. It is not difficult to see that

$$\mathcal{F}_1 = [\{f_T : T \in \mathcal{T}_1\}] = [\{f_T : T \in \mathcal{T}_3\}],$$
$$\mathcal{F}_2 = [\{f_T : T \in \mathcal{T}_2\}] = [\{f_T : T \in \mathcal{T}_4\}].$$

Using Lemma 5 we conclude that the classes $\mathcal{F}_1$ and $\mathcal{F}_2$ are not α-bounded.

Theorem 3 *Let $\mathcal{F}$ be a closed class of ROFs. Then $\mathcal{F}$ is α-bounded if and only if $\mathcal{F}$ contains none of the classes $\mathcal{F}_1, \mathcal{F}_2$.*

Proof Let $\mathcal{F}$ be α-bounded. Then $\mathcal{F}$ contains none of the classes $\mathcal{F}_1, \mathcal{F}_2$ since these classes are not α-bounded.

Let $\mathcal{F}$ contain none of the classes $\mathcal{F}_1, \mathcal{F}_2$. We now show that $\mathcal{F}$ is α-bounded. Assume the contrary: $\mathcal{F}$ is not α-bounded. Denote by $\mathcal{T}$ the set of read-once trees containing for each function $g \in \mathcal{F}$ all trees T such that $f_T = g$. It is clear that the set $\mathcal{T}$ is closed under the operation of renaming of labels. Since the class $\mathcal{F}$ is closed under the operation of renaming of variables, the set $\mathcal{T}$ is closed under the operation of renaming of variables. Using Remark 1, we obtain that $\mathcal{T}$ is closed under the operation of removal of variables. Therefore $\mathcal{T}$ is a closed class of ROTs. Since $\mathcal{F}$ is not α-bounded, the class $\mathcal{T}$ is not α-bounded. By Lemma 6, $\mathcal{T}$ contains at least one of the classes $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4$. Therefore, $\mathcal{F}$ contains one of the classes $\mathcal{F}_1, \mathcal{F}_2$. This contradicts our initial assumption and completes the proof of the theorem. □

4.3 Parameter β

In this section, we study the parameter $\beta(f)$ that, for any Boolean function f with n relevant variables, is defined in the following way:

$$\beta(f) = n/C(f).$$

To characterize the parameter β in the terminology of minimal closed classes of read-once functions, which are not β-bounded, first we make the following observation, which follows directly from the definition of certificate complexity.

Claim 2 *$C(T') \leq C(T)$ for any subtree T' of a read-once tree T.*

We also quote the following result, which was proved in [27] (see Theorem 3) in the terminology of read-once functions.

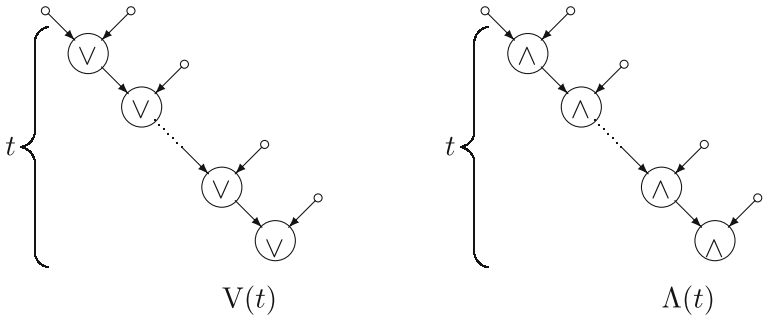

Fig. 3 Partially labelled ROTs $\mathrm{V}(t)$ and $\Lambda(t)$

Lemma 7 *If T is a read-once tree with n inputs, then*

$$n \leq C_0(T)C_1(T).$$

Now we denote by $\mathrm{V}(t)$ and $\Lambda(t)$ partially labelled read-once trees corresponding to disjunction and conjunction of t variables, respectively, see Fig. 3, and prove a lemma concerning read-once trees that do not contain either $\mathrm{V}(t)$ or $\Lambda(t)$ as subtrees.

Lemma 8 *If T is a read-once tree with n inputs that does not contain $\mathrm{V}(t)$ or $\Lambda(t)$, then*

$$n \leq 2^t C(T).$$

Proof We prove the result for $\Lambda(t)$, since for $\mathrm{V}(t)$ the proof is similar. Let v be the output of T and let v_1, v_2 be the parents of v. We consider two subtrees T_1 and T_2 of T with outputs v_1 and v_2, respectively, and denote the number of inputs in these trees by n_1 and n_2, respectively.

If v is a $\vee$-gate, then applying Lemmas 2, 4 and 7, we obtain

$$\begin{aligned} n = n_1 + n_2 &\leq C_0(T_1)C_1(T_1) + C_0(T_2)C_1(T_2) && \text{by Lemma 7} \\ &\leq 2^t(C_0(T_1) + C_0(T_2)) && \text{by Lemma 4} \\ &\leq 2^t \max\{C_0(T_1) + C_0(T_2), \max\{C_1(T_1), C_1(T_2)\}\} \\ &= 2^t C(T) && \text{by Lemma 2.} \end{aligned}$$

If v is a $\wedge$-gate, then $h_\wedge(T_1) \leq t-1$ and $h_\wedge(T_2) \leq t-1$, since otherwise T contains $\Lambda(t)$. Applying Lemmas 2, 4 and 7, we obtain

$$\begin{aligned} n = n_1 + n_2 &\leq C_0(T_1)C_1(T_1) + C_0(T_2)C_1(T_2) && \text{by Lemma 7} \\ &\leq 2^{t-1}(C_0(T_1) + C_0(T_2)) && \text{by Lemma 4} \\ &\leq 2^{t-1} 2 \max\{C_0(T_1), C_0(T_2)\} \\ &\leq 2^t \max\{\max\{C_0(T_1), C_0(T_2)\}, C_1(T_1) + C_1(T_2)\} \\ &= 2^t C(T) && \text{by Lemma 2.} \end{aligned}$$

□

To prove the main result of this section we denote by $\Gamma_1(p,t)$ and $\Gamma_2(p,t)$ partially labelled read-once trees represented in Fig. 4. Also, let

$$\mathcal{P}_1 = [\{\Gamma_1(p,t) \ : \ p,t = 0,1,2\ldots\}]$$

$$\mathcal{P}_2 = [\{\Gamma_2(p,t) \ : \ p,t = 0,1,2\ldots\}].$$

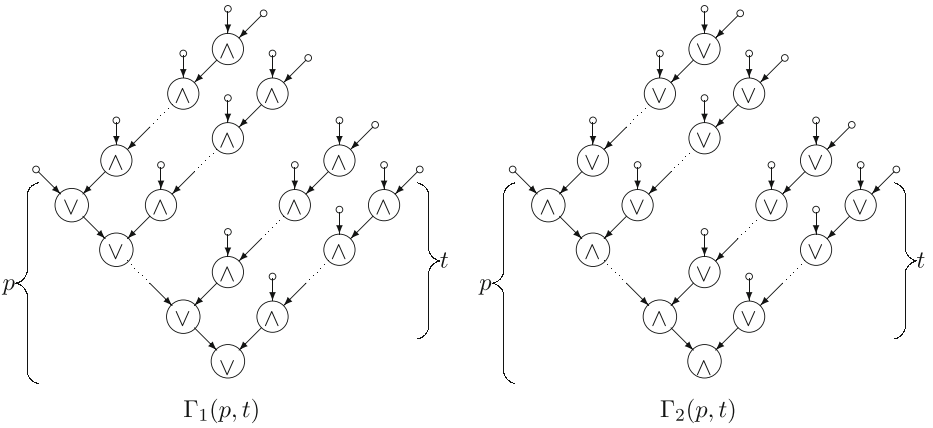

Fig. 4 Partially labelled ROTs $\Gamma_1(p,t)$ and $\Gamma_2(p,t)$

Theorem 4 *Let $\mathcal{P}$ be a closed class of read-once trees. Then $\mathcal{P}$ is β-bounded if and only if $\mathcal{P}$ contains neither $\mathcal{P}_1$ nor $\mathcal{P}_2$.*

Proof Let f_n be a read-once function represented by a read-once tree $\Gamma_1(\sqrt{n},\sqrt{n})$ with $n+\sqrt{n}+1$ inputs. Then $\beta(f_n) = n/(\sqrt{n}+1)+1$ and hence the parameter β is unbounded in the class $\mathcal{P}_1$. Similarly, it is unbounded in the class $\mathcal{P}_2$, which proves the "only if" part of the theorem.

To prove the "if" part, we fix natural numbers p, q, t and show that if T is a read-once tree with n inputs containing neither $\Gamma_1(p,t)$ nor $\Gamma_2(q,t)$, then

$$n \le C(T)2^{2(p+q+t)}.$$

Let v be the output of T and v_1, v_2 the parents of v. We consider two subtrees T_1 and T_2 of T with outputs v_1 and v_2, respectively.

Case 1: Assume that either

(a) v is a $\vee$-gate and neither T_1 nor T_2 contains $\Lambda(t)$, or
(b) v is a $\wedge$-gate and neither T_1 nor T_2 contains $\mathrm{V}(t)$.

In case 1(a) T does not contain $\Lambda(t)$ and in case 1(b) T does not contain $\mathrm{V}(t)$. In both cases, $n \le 2^t C(T)$ by Lemma 8.

Case 2: Assume that either

(a) v is a $\vee$-gate and both T_1 and T_2 contain $\Lambda(t)$, or
(b) v is a $\wedge$-gate and both T_1 and T_2 contain $\mathrm{V}(t)$.

In case 2(a), we conclude that neither T_1 nor T_2 contains $\Gamma_1(p-1,t)$, since otherwise T contains $\Gamma_1(p,t)$. Now, applying induction on $p+q$ we obtain

$$n = n_1+n_2 \le C(T_1)2^{2(p-1+q+t)}+C(T_2)2^{2(p-1+q+t)} \le 2C(T)2^{2(p-1+q+t)} \le C(T)2^{2(p+q+t)}.$$

Case 2(b) implies the same conclusion by similar arguments.

Case 3: Assume that either

(a) v is a $\vee$-gate, T_1 contains $\Lambda(t)$ and T_2 does not contain $\Lambda(t)$, or
(b) v is a $\wedge$-gate, T_1 contains $\mathrm{V}(t)$ and T_2 does not contain $\mathrm{V}(t)$.

Then we define $T' := T_1$ and apply to T' the above case analysis iteratively as long as we are in Case 3. When we encounter either Case 1 or Case 2, we stop.

Assume first that the analysis has stopped, when we encountered Case 1. Then the original tree T has been partitioned into subtrees of two types: subtrees that do not contain $\Lambda(t)$ and subtrees that do not contain $\mathrm{V}(t)$. The output of every subtree of the first type has a $\vee$-gate as a child. Therefore, the union of all these subtrees, which we denote by $T_\vee$ and which is a tree obtained from T by removing all inputs from the subtrees of the second type, does not contain $\Lambda(t)$. Similarly, the output of every subtree of the second type has a $\wedge$-gate as a child, and hence the union of all these subtrees, denoted $T_\wedge$, does not contain $\mathrm{V}(t)$. Denoting the number of inputs in $T_\vee$ and $T_\wedge$ by $n_\vee$ and $n_\wedge$ respectively, we obtain

$$n = n_\vee + n_\wedge \leq C(T_\vee)2^t + C(T_\wedge)2^t \leq 2C(T)2^t \leq C(T)2^{2(p+q+t)}.$$

Finally, assume that the analysis has stopped at Case 2. Then the tree T has been partitioned into subtrees $T_\vee$ and $T_\wedge$, defined in the previous paragraph, and two subtrees T_1' and T_2' that appear in Case 2. Assuming, without loss of generality that neither T_1' nor T_2' contains $\Gamma_1(p-1,t)$, we conclude by induction on $p+q$ that

$$\begin{aligned} n = n_1' + n_2' + n_\vee + n_\wedge &\leq C(T_1')2^{2(p-1+q+t)} + C(T_2')2^{2(p-1+q+t)} + C(T_\vee)2^t + C(T_\wedge)2^t \\ &\leq 4C(T)2^{2(p-1+q+t)} \\ &\leq C(T)2^{2(p+q+t)}. \end{aligned}$$

□

In order to translate Theorem 4 to the language of read-once functions, we denote

$$g_1(p,t) = \bigvee_{i=1}^{p}(x_{(i-1)t+1} \wedge x_{(i-1)t+2} \wedge \ldots \wedge x_{(i-1)t+t}),$$

$$g_2(p,t) = \bigwedge_{i=1}^{p}(x_{(i-1)t+1} \vee x_{(i-1)t+2} \vee \ldots \vee x_{(i-1)t+t}).$$

Also,

$$\mathcal{G}_1 = [\{g_1(p,t) \ : \ p,t = 0,1,2\ldots\}],$$
$$\mathcal{G}_2 = [\{g_2(p,t) \ : \ p,t = 0,1,2\ldots\}].$$

Theorem 5 *Let $\mathcal{G}$ be a closed class of read-once functions. Then $\mathcal{G}$ is β-bounded if and only if $\mathcal{G}$ contains neither $\mathcal{G}_1$ nor $\mathcal{G}_2$.*

5 Concluding Remarks and Open Problems

In this paper, we defined a quasi-order on the set of read-once Boolean functions and showed that this is a well-quasi-order. This conclusion defines a quasi-order on the set of parameters that measure complexity of read-once functions and provides a uniform way of describing parameters through critical classes, i.e. minimal classes where the parameters jump to infinity.

We observed in the paper that a number of important complexity measures, such as certificate complexity or deterministic decision tree complexity, are equivalent in the universe of read-once Boolean functions and lie at the bottom of the hierarchy of parameters. To extend the hierarchy, we have introduced two new parameters α and β related to certificate complexity and characterized both of them by means of critical classes, showing that α lies

strictly above certificate complexity and β lies strictly above α. We believe that the new parameters will find applications in various aspects of the analysis of Boolean functions, such as, for instance, parameter-efficient learning of Boolean functions [10].

There is a variety of other parameters that measure complexity of Boolean functions, such as submodular goal value [5], average sensitivity [34], various nonlinearity measures [7], etc. Characterizing them through critical classes remains a challenging open problem.

Acknowledgements Research reported in this publication was supported by the King Abdullah University of Science and Technology (KAUST).

The authors are greatly indebted to anonymous reviewers for useful comments and suggestions, many of which have been incorporated in the text.

References

1. Albert, M.H., Atkinson, M.D., Vatter, V.: Subclasses of the separable permutations. Bull. Lond. Math. Soc. **43**(5), 859–870 (2011)
2. Angluin, D., Hellerstein, L., Karpinski, M.: Learning read-once formulas with queries. J. ACM **40**, 185–210 (1993)
3. Aschenbrenner, M., Hemmecke, R.: Finiteness theorems in stochastic integer programming. Found. Comput. Math. **7**, 183–227 (2007)
4. Atminas, A., Brignall, R.: Well-quasi-ordering and finite distinguishing number. J. Graph Theory **95**, 5–26 (2020)
5. Bach, E., Dusart, J., Hellerstein, L., Kletenik, D.: Submodular goal value of Boolean functions. Discrete Appl. Math. **238**, 1–13 (2018)
6. Alecu, B., Atminas, A., Lozin, V., Zamaraev, V.: Graph classes with linear Ramsey numbers. Discrete Math. **344**, 112307 (2021)
7. Boyar, J., Find, M.G., Peralta, R.: On various nonlinearity measures for boolean functions. Cryptogr. Commun. **8**(3), 313–330 (2016)
8. Brignall, R., Korpelainen, N., Vatter, V.: Linear clique-width for hereditary classes of cographs. J. Graph Theory **84**, 501–511 (2017)
9. Buhrman, H., de Wolf, R.: Complexity measures and decision tree complexity: a survey. Theoret. Comput. Sci. **288**, 21–43 (2002)
10. Bystrygova, A.V.: Parameter-efficient learning of Boolean functions from closed post classes. (Russian) Diskret. Mat. **31**, 34–57 (2019)
11. Choi, I., Furuya, M., Kim, R., Park, B.: A Ramsey-type theorem for the matching number regarding connected graphs. Discrete Math. **343**, 111648 (2020)
12. Couceiro, M., Pouzet, M.: On a quasi-ordering on Boolean functions. Theoret. Comput. Sci. **396**, 71–87 (2008)
13. Crama, Y., Hammer, P.L.: Boolean Functions. Theory, Algorithms, and Applications. Encyclopedia of Mathematics and its Applications, 142. Cambridge University Press, Cambridge. xxii+687 pp. ISBN: 978-0-521-84751-3 (2011)
14. Fishburn, P.C.: An interval graph is not a comparability graph. J. Combin. Theory **8**, 442–443 (1970)
15. Ganian, R., Hliněný, P., Nešetřil, J., Obdržálek, J., Ossona de Mendez, P.: Shrub-depth: capturing height of dense graphs. Log. Methods Comput. Sci. **15** (2019), no. 1, Paper No. 7, 25 pp. https://lmcs.episciences.org/5149

16. Golumbic, M.C., Mintz, A., Rotics, U.: Factoring and recognition of read-once functions using cographs and normality and the readability of functions associated with partial k-trees. Discrete Appl. Math. **154**(10), 1465–1477 (2006)
17. Gurvich, V.A.: On the normal form of positional games. (Russian) Dokl. Akad. Nauk SSSR **264**(1), 30–33 (1982)
18. Huang, H.: Induced subgraphs of hypercubes and a proof of the sensitivity conjecture. Ann. of Math. (2) **190**(3), 949–955 (2019)
19. Jean, M.: An interval graph is a comparability graph. J. Combin. Theory **7**, 189–190 (1969)
20. Karchmer, M., Linial, N., Newman, I., Saks, M., Wigderson, A.: Combinatorial characterization of read-once formulae. Discrete Math. **114**, 275–282 (1993)
21. Kauffman, S., Peterson, C., Samuelsson, B., Troein, C.: Random Boolean network models and the yeast transcriptional network. Proc. Natl. Acad. Sci. **100**(25), 14796–14799 (2003)
22. Kruskal, J.B.: Well-quasi-ordering, the tree theorem, and Vazsonyi's conjecture. Trans. Am. Math. Soc. **95**, 210–225 (1960)
23. Laver, R.: On fraïsse's order type conjecture. Ann. Math. **93**(1), 89–111 (1971)
24. Lozin, V.: Graph parameters and Ramsey theory. Lecture Notes in Comput. Sci. **10765**, 185–194 (2018)
25. Lozin, V., Milanic, M.: Critical properties of graphs of bounded clique-width. Discrete Math. **313**, 1035–1044 (2013)
26. Lozin, V., Razgon, I., Zamaraev, V., Zamaraeva, E., Zolotykh, N.: Linear read-once and related Boolean functions. Discrete Appl. Math. **250**, 16–27 (2018)
27. Morizumi, H.: Sensitivity, block sensitivity, and certificate complexity of unate functions and read-once functions. In: IFIP TCS. Lecture Notes in Comput. Sci., vol. 8705, pp. 104–110 (2004)
28. Moshkov, M.: About the depth of decision trees computing Boolean functions. Fundam. Inform. **22**(3), 203–215 (1995)
29. Moshkov, M.: Time complexity of decision trees. Transactions on Rough Sets III. Lecture Notes in Comput. Sci. **3400**, 244–459 (2005)
30. Moshkov, M.: Comparative analysis of deterministic and nondeterministic decision trees. Intelligent Systems Reference Library, 179. Springer, Cham (2020). xvi+297 pp. ISBN: 978-3-030-41727-7 https://www.springer.com/gp/book/9783030417277
31. Nash-Williams, C.St.J.A.: On well-quasi-ordering infinite trees. Proc. Camb. Phil. Soc. **61**, 697–720 (1965)
32. Robertson, N., Seymour, P.D.: Graph minors. V. Excluding a planar graph. J. Combin. Theory Ser. B **41**, 92–114 (1986)
33. Robertson, N., Seymour, P.D.: Graph minors. XX. Wagner's conjecture. J. Combin. Theory Ser. B **92**, 325–357 (2004)
34. Stearns, R.E., Rosenkrantz, D.J., Ravi, S.S., Marathe, M.V.: A characterization of nested canalyzing functions with maximum average sensitivity. Discrete Appl. Math. **251**, 5–14 (2018)
35. Stetsenko, V.: On almost bad Boolean bases. Theoret. Comput. Sci. **136**, 419–469 (1994)
36. Tao, J.: Pattern occurrence statistics and applications to the Ramsey theory of unavoidable patterns. Online J. Anal. Comb. **13**, 22 (2018)
37. Wang, C.: Boolean minors. Discrete Math. **141**, 237–258 (1995)
38. Zverovich, I.E.: Characterizations of closed classes of Boolean functions in terms of forbidden subfunctions and Post classes. Discrete Appl. Math. **149**, 200–218 (2005)