Simple Infeasibility Certificates for Attack Trees

Ahto Buldas[1,2] and Aleksandr Lenin[1,2,3] and Jan Willemson[1,3] and Anton Charnamord[2] ⋆

[1] Cybernetica AS, Mäealuse 2/1, Tallinn, Estonia
[2] Tallinn University of Technology, Ehitajate tee 5, Tallinn, Estonia
[3] Software Technology and Applications Competence Centre, Tallinn, Estonia

Abstract. We introduce infeasibility certificates, compact and easily verifiable proofs that no profitable attacks exist in the considered system model. We introduce computational methods for generation and validation of such proofs using an enhanced weight reduction technique. A new method for obtaining adversarial expenses by approximating an interval within which this value resides, is an interesting approach to tackle NP-complete tasks and allows to obtain values that require extensive computations in reasonable time.

1 Introduction

Attack trees are regularly used to perform cost-benefit analysis [2, 15, 11, 13, 4, 3] to determine if the considered system model is sufficiently protected from rational profit-oriented adversaries. In these models, the adversarial profit as well as the expenses, related to preparing and launching single attack steps, are known to adversaries. If the profit exceeds expenses, the considered system is vulnerable, as it is profitable to attack it. The objective of the defender is to deploy security measures in a way that will make the attack process unprofitable for attackers. According to the rationality assumption, non-profitability of attacking is considered to be sufficient to hold rational attackers away from the considered system.

An attack tree is a hierarchical description of targeted attacks driven by a common goal. It is a tree-like structure where every leaf corresponds to an elementary attack step, annotated with corresponding cost to prepare and launch the attack step. In a more formal setting, an attack tree is a monotone Boolean function $\Phi(x_1, x_2, \ldots, x_n)$, the arguments of which are assigned with their corresponding weights (costs). The goal of an attacker is to satisfy Φ in the cheapest possible way. We denote by $w(\Phi)$ the minimal cost required to satisfy Φ.

Existing analysis techniques like the improved failure-free model [3], approach the problem from the adversarial point of view, providing evidence about feasibility of attacking by searching for the cheapest satisfying assignment to Φ the

⋆ The research leading to these results has received funding from the European Regional Development Fund through Estonian Centre of Excellence in ICT Research (EXCITE) and the Estonian Research Council under Institutional Research Grant IUT27-1.

total weight of which does not exceed a certain threshold T. In this paper we study the same problem from the defender's perspective and aim at providing evidence about infeasibility of attacking, which is equivalent to showing that no profitable attacks exist in the model. In other words, we need to show that $w(\Phi) > K$, where K is the adversarial profit.

Calculating $w(\Phi)$ is computationally expensive, as we need to consider every possible solution to find the cheapest one. We take a different approach and approximate the interval within which $w(\Phi)$ is determined. This provides us with a reasonable approximation of $w(\Phi)$ and lets us obtain the result in a much more efficient way, compared to calculating $w(\Phi)$ directly. We obtain this interval by approximating the upper $w_u(\Phi)$ and lower $w_l(\Phi)$ bounds of $w(\Phi)$ using genetic algorithms.

If $w_l(\Phi) > K$, attacking is infeasible, and this is what we need to achieve by deploying security measures. On the contrary, when $w_u(\Phi) < K$ attacking is profitable. If K lies somewhere between $w_l(\Phi)$ and $w_u(\Phi)$, it is not possible to make reliable conclusions about feasibility of attacking, this situation would require more detailed analysis. Relying just on $w_l(\Phi)$ is insufficient – we need to obtain both bounds of $w(\Phi)$ to be able to estimate the relative error $\delta = \frac{w_u(\Phi)-w_l(\Phi)}{w_u(\Phi)}$ of the method. If the relative error is big then it might be that security expenses are way too high. The less the relative error is, the less unnecessary investments we probably make. We study how well $w_l(\Phi)$ approximates $w(\Phi)$ given a specific computational method that computes $w_l(\Phi)$ for the Boolean function Φ. The better it does, the more precise estimation of $w(\Phi)$ we can obtain. It turned out that there are cases when $w_l(\Phi)$ equals $w(\Phi)$ so that the estimation is exact, but this is not always the case. Apart from the computational method presented in this paper, there exist other methods which can be used to obtain $w_l(\Phi)$, e.g. [8].

We need to be able to prove to the auditors that the values $w_u(\Phi)$ and $w_l(\Phi)$ are correctly calculated. Given $w_u(\Phi)$, it is easy to prove that this value is indeed an upper bound. To prove it, it is sufficient to present any satisfying solution with weight $w_u(\Phi)$, as there exists efficient algorithm that calculates the weight of a given solution. For $w_l(\Phi)$, in general, such a compact proof does not exist.

Therefore we introduce infeasibility certificates that are compact and easily verifiable proofs ξ that the number $w_l(\Phi)$ is indeed a lower bound of the unknown value $w(\Phi)$. In practice, all we need to do in order to verify the feasibility of attacking, is to calculate the value of $w_l(\Phi)$ from the Boolean function Φ and a certificate ξ, and compare the obtained value to the adversarial profit K. If $w_l(\Phi) > K$, attacking is unprofitable. Infeasibility certificates may be used, for instance, to show auditors that the system is reasonably protected against rational profit-oriented adversaries. Such a certificate provides a proof that can be verified in a very time-efficient manner, even though the generation of such a certificate may be computationally expensive and take several days.

The outline of this paper is the following. Section 2 outlines the state of the art, Section 3 introduces efficient way to calculate adversarial expenses by means of the weight reduction method. Section 4 introduces infeasibility certificates, and their usage is demonstrated in Section 5 on a simple example. Section 6 provides relevant theorems and proofs. Section 7 outlines empirical evidence collected throughout the experiments studying the quality of approximation. Section 8 discusses some open questions.

2 Notation and Definitions

Let $X = \{x_1, x_2, \ldots\}$ be a fixed set of independent Boolean variables. For Boolean disjunction and conjunction operations, we use the conventional addition $(x_1 + x_2)$ and multiplication $x_1 x_2$, respectively. We use the notation $\Phi(X)$ to denote a Boolean function that may depend on any finite set of the Boolean variables in X. By $\Phi(x_1, x_2, \ldots, x_n)$, we mean a Boolean function that may depend on the variables $x_1, x_2, \ldots, x_n$, but not on any other variable in X. By a *conjunction* we mean any conjunction of a finite set of Boolean variables in X; for example $x_{i_1} x_{i_2} \ldots x_{i_k}$. By a *min-term* of a boolean function $\Phi(X)$, we mean any conjunction μ which implies the truth of Φ, i.e. for which $\mu \models \Phi(X)$. By a weight function w, we mean any function $w\colon X \to \mathbb{R}^+$ which for any variable $x \in X$ returns a non-negative real number $w(x)$. Weight functions can be extended to all Boolean functions as follows:

- $w(\mu) = w(x_1) + \ldots + w(x_n)$, if $\mu = x_1 x_2 \ldots x_n$;
- $w(\Phi) = \min\{w(\mu)\colon\ \mu \text{ is a min-term of } \Phi\}$ for any Boolean function Φ.

A min-term μ of Φ is a *cheapest min-term* if $w(\mu) = w(\Phi)$. The question of whether there exist feasible (to adversary) attacks against a system can be formalised in terms of the Weighted Monotone Satisfiability (WMSAT) problem.

Definition 1. *The* WMSAT *language consists of triples* (Φ, w, t)*, where* Φ *is a monotone Boolean formula (AND/OR-formula),* w *is a function that to every variable in* Φ *associates a weight* $w(z)$*, and* t *is a real number (threshold) so that there is a min-term* μ *of* Φ *with* $w(\mu) \leq t$*, where* $w(\mu)$ *is the sum of all* $w(z)$ *such that* z *is a variable in* μ*.*

In practical context, $X = (\Phi, w, t)$ represents possible attacks against the system (via the monotone Boolean function Φ), their costs (via the weight function w), as well as the estimated income of an adversary (via the threshold value t). It is known that the WMSAT problem is NP-complete [3, 14]:

Theorem 1. *The Weighted Monotone Satisfiability Problem is NP-complete.*

Proof. We will show that the Vertex Cover problem can be polynomially reduced to the WMSAT problem. Let $\mathcal{G}$ be the graph with a vertex set $\{v_1, \ldots, v_m\}$. We define a Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ as follows. For each edge (v_i, v_j) of $\mathcal{G}$ we define the clause $\mathcal{C}_{i_j} = x_i \vee x_j$. The Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ is defined as the conjunction of all $\mathcal{C}_{i_j}$ such that (v_i, v_j) is an edge of $\mathcal{G}$. Let the weight w_i of each x_i be equal to 1. It is obvious that $\mathcal{G}$ has a vertex cover $\mathcal{S}$ of size $|\mathcal{S}| < \mathcal{P}$ iff the monotone Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ has a satisfying assignment with a total weight less than $\mathcal{P}$. $\square$

For any language $L \in \mathbf{NP}$, there is an efficient (poly-time) verification algorithm $V(X, \mu)$, such that for every X $X \in L \Leftrightarrow \exists\mu\colon V(X, \mu) = 1$. From the view-point of attack-trees and security, what one really wants is a proof ξ (of size polynomial in $|X|$) that $X \notin \mathsf{WMSAT}$, which in the attack tree language means that there are no attacks with the cost less (or equal) than t. So, ideally we would like to have a poly-time verification algorithm W such that for every $X = (\Phi, w, t)$ $X \notin \mathsf{WMSAT} \Leftrightarrow \exists\xi\colon W(X, \xi) = 1$, but the existence of such W would imply $\mathbf{NP} = \mathbf{coNP}$ which, as the complexity theorists tend to believe, is not true. Hence, it is a very little hope to find such a W. We still may have an efficient verifier W that works for some instances X, i.e. we only have the right-to-left implication.

Definition 2. *A* partial infeasibility verifier *for a language L is a poly-time computable function W such that for every X : $X \notin L \Leftarrow \exists\alpha\colon W(X, \alpha) = 1$. Any α such that $W(X, \alpha) = 1$ is called an* infeasibility certificate *for X.*

In the practical attack-tree context, $L = \mathsf{WMSAT}$ and α is a proof that X is not in WMSAT which in practical context means that there exist no feasible attacks against the system. Practical usability of W depends on for how many useful X-s the function W works, as well as on the size of α.

3 Related Work

3.1 Attack Tree Semantics

The idea of analyzing security using the so-called attack trees was popularized by Schneier in [18], who suggested to use attack trees as a convenient hierarchical representation of an attack scenario. The model of Buldas *et al.* [2] is remarkable for introducing the multi-parameter approach to the quantitative security risk analysis. The model assumes rational adversaries who behave fully adaptively and are always trying to maximize their average outcome. The authors state that in order to assess security it is sufficient to assess adversarial utility. Their model introduced a novel way to think about security and gave start to multi-parameter quantitative security analysis. Jürgenson and Willemson have shown that Buldas *et al.* model is inconsistent with Mauw-Oostijk foundations [15] and introduced the so-called parallel model [11] and the serial model [12] which provided more reliable results, however in neither models the adversary behaved in a fully adaptive way. Also, these models were intractable from the computational point of view, which is why Jürgenson and Willemson developed genetic approximation methods [13]. All the previously described models tried to approach the problem of estimating adversarial expenses from above considering only the upper bound of adversarial expenses. Buldas-Stepanenko fully adaptive model [4] was the first model which tried to take a different view on the problem and aimed at estimating adversarial expenses from below considering lower bounds of expenses. Their failure-free model is similar to the fully adaptive model with the only difference that in the failure-free model success probabilities of the attack steps are equal to 1. The most significant contribution of the paper [4] is the upper bounds ideology by which the models should estimate adversarial utility from above, trying to avoid false-positive security results. The improved failure-free model [3] improves the Buldas-Stepanenko failure-free model [4] by eliminating the force-failure states. In the improved model the adversarial behavior more fully conforms to the upper bounds ideology introduced in [4]. It turned out that the elimination of the force failure states has made the model computationally easier. The authors show that finding an optimal strategy in the new model is NP-complete and introduced the cost reduction and propagation methods to get an estimation of the lower bound of adversarial expenses.

3.2 Infeasibility Certificates

Here we list several possible methods that could be used to certify infeasibility of a WMSAT instance.

There exist satisfiability modulo theories (SMT) solvers, like Z3 [16], which can generate proofs of unsolvability for infeasible SMT instances. A WMSAT instance may be encoded as an SMT instance and an infeasibility proofs may be obtained from the solvers, however these proofs cannot be easily verified and require proof assistants, such as Coq [7], to verify them. Additionally, the performance of solving WMSAT instances encoded as SMT instances is questionable. Infeasibility of a system of linear equations may be certified by an infeasibility certificate the existence of which follows from the Range/kernel theorem [5] for complex domain and from the Farkas' lemma [6] for the real domain. We can represent the WMSAT problem in the form of a system of linear equations with real coefficients and theoretically generate an infeasibility certificate for it. However, in practice this approach is inefficient, as this involves converting an arbitrary propositional Boolean formula Φ into CNF, which is computationally expensive, and besides that, the size of the matrix A of coefficients of linear constraint inequalities ($A \cdot x \geqslant b$ in the canonical representation of LP) grows exponentially in the number of variables in Φ. Two notable results from semialgebraic geometry and convex optimization, Hilbert's Nullstellensatz [19] and Stengle's Positivstellensatz [20, 21], state that for every infeasible system of polynomial equations and inequalities there exist a simple algebraic identity that directly certifies the infeasibility of the system. Nullstellensatz deals with polynomial equations in complex variables, and has been studied for the 3-colorability problem [8]. Nullstellensatz seems not to fit well for certifying infeasibility of a WMSAT instance, as it cannot be encoded as a system of polynomial equations. Positivstellensatz deals with arbitrary systems of polynomial equations and inequalities and therefore is a promising approach. Generating a Positivstellensatz infeasibility certificate is a convex feasibility problem. Finding bounded degree solutions to the Positivstellensatz is a semideinite problem. This gives a hierarchy of syntactically verifiable certificates, the validity of which may be easily checked. However, solving semidefinite programs (SDPs) is NP-hard for quartic and higher degree polynomials [1, 10]. A computational tractable replacement for this is to apply sum-of-squares (SOS) relaxation to obtain an optimization problem over affine families of polynomials, subject to SOS constraints. It is known that SOS programs may be solved in polynomial time [9, 17]. This approach provides a computational method to generate a bounded degree Positivstellensatz infeasibility certificates for given WMSAT instances in polynomial time, and is therefore a promising solution, which will be considered for future research. In this paper we focus on computational methods for generating and verifying infeasibility certificates for WMSAT instances, which use attack trees and does not require any conversions between attack trees and other equivalent representations, e.g. systems of polynomials.

4 Efficient adversarial expenses calculation

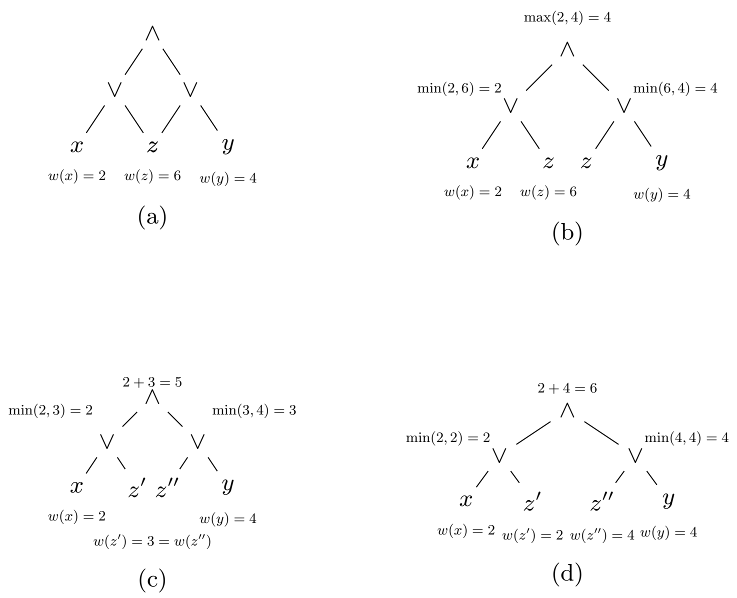

Fig. 1. Adversarial expenses calculation: an attack tree (a), propagation method (b), cost reduction (c) and (d).

An attack tree shown on the left-hand side in Fig. 1a corresponds to the Boolean function $\phi(x, y, z) = (x+z)(z+y)$, where logical conjunction is denoted as multiplication, and logical disjunction is denoted as sum. Each argument of the function is annotated with a weight. Let the arguments have the following weights: $w(x) = 2, w(z) = 6$, and $w(y) = 4$.

In order to find $w(\phi)$ we need to find the weight of the cheapest satisfying assignment. ϕ has five min-terms: $(xz), (zy), (xy), (xzy)$, and (z). Hence, $w(\phi) = \min\{w(xz), w(zy), w(xy), w(xzy), w(z)\} = \min\{8, 10, 6, 12, 6\} = 6$. Calculating the weight of the function this way is computationally expensive for large trees and not quite practical. The tree structure of an attack tree facilitates the usage of the efficient propagation methods [3, 14] that allow to calculate $w_l(\phi)$ in time $\mathcal{O}(n)$, where n is the number of attack tree nodes.

Definition 3. *A propagation method is a pair* $(\bigwedge, \bigvee)$ *of real-valued binary operations such that for every two Boolean functions* F *and* G*:*

$$\bigwedge(w(F), w(G)) \leqslant w(FG) \quad \textit{and} \quad \bigvee(w(F), w(G)) \leqslant w(F+G)$$

For instance, the pair $(\max, \min)$ is a propagation method, because

$$\max(w(F), w(G)) \leqslant w(FG) \quad \text{and} \quad \min\{w(F), w(G)\} = w(F+G)$$

However, it can be seen in Fig 1b that the max operator is a very rough estimation of $w(\phi)$ and one could think of using $(+, \min)$ instead. The problem is that this is not a propagation method because it is possible that $w(F)+w(G) > w(FG)$. However, we are able to convert the function FG into a function $F'G'$ for which $(+, \min)$ is a propagation method in terms of Def. 3. This can be done by using the technique known as weight reduction [3, 14]. The rule $w(F) + w(G) \leqslant w(FG)$ holds only if F and G have no common variables. The weight reduction technique eliminates the common variables in the following way. If z is a common variable of F and G, then we first replace z with new independent variables z' in F and z'' in G, and define the weights so that $w(z') + w(z'') = w(z)$. It can be shown that $w(F'G') \leqslant w(FG)$ according to Thm. 4.5.5 in [14]. The functions F' and G' now have one less common variable, and we can repeat the procedure with other common variables.

As the Boolean function in Fig. 1b contains a common variable z, and therefore the propagation method $(+, \min)$ cannot be applied directly, and weight reduction must be done first. The common variable z is substituted with its independent copies $z \mapsto z'$ and $z \mapsto z''$, and their corresponding weights are distributed in a way that $w(z') + w(z'') = w(z)$. This distribution may be done in a variety of ways, as illustrated in Fig. 1c and 1d, both representing the reduced function $\phi(x, z', z'', y) = (x + z')(z'' + y)$. Independently on how we distribute weights, the rule $w(z') + w(z'') = w(z)$ must hold. In the case of a symmetric distribution $w(z') = 3 = w(z'')$ the result is underestimated, as can be seen in Fig. 1c, however, asymmetric distribution $w(z) = 2$ and $w(z'') = 4$ produces the exact result for $w(\phi)$ as can be seen in Fig. 1d. Thus, the choice of a distribution sets the precision with which the exact result can be approximated.

The main idea behind weight reduction is that every instance of the common variable in a Boolean formula is substituted with an independent copy of this variable with reduced weight. The sum of the weights of the reduced variables must never exceed the initial weight of the common variable, thus guaranteeing that this method will not over-estimate the weight of the function, according to Thm. 4.5.5 in [14]. We will show that in the case of Boolean functions with 1 common variable, the result is always exact. We refer the reader to Thm. 4 for details. If the Boolean function contains 2 or more common variables, this does not hold due to the existence of the so-called reduction defect. For example, the function $(vc + auz)(uy + bz)$ with unit costs for its variables, always has a reduction defect.

4.1 Infeasibility Certificates for WMSAT

In this subsection we present a method of constructing infeasibility certificates for WMSAT that uses weight reduction and propagation described above. The certificate α consists of instructions for a complete weight reduction of Φ. For any conjunction-type sub-formula $\Psi = FG$ of Φ for any common variable z of F and G we have to describe how we distribute the weight of z between F and G. For this, we have to define a real number $\alpha_{\Psi,z} \in [0,1]$ which means that if the weight of z was w, then it will be redefined to αw in F and $(1-\alpha)w$ in G. We call a pair (Ψ, z) a *decision point* of Φ. All the values $\alpha_{\Psi,z}$ together form a *certificate*. Formally, the certificate is a function α that assigns to each decision point (Ψ, z) of Φ a real number $\alpha(\Psi, z)$ from the unit interval $[0,1]$.

Definition 4. *By a* decision point *of a Boolean formula Φ we mean a pair (Ψ, z) where Ψ is a conjunction-type sub-formula FG of Φ so that z is a common variable of F and G. The set of all decision points of Φ is denoted by $\mathcal{D}(\Phi)$.*

Definition 5. *By a* certificate *for a Boolean formula Φ we mean a function $\alpha\colon \mathcal{D}(\Phi) \to [0,1]$.*

As described above, the certificate consists of instructions how to weight-reduce (Φ, w) to (Φ', w') so that the propagation method $(+, \min)$ computes the exact value of $w'(\Phi')$ that of course may be smaller than $w(\Phi)$.

4.2 Computational Methods

The algorithm Desp that for a Boolean formula Φ computes the list $\mathcal{D} = \mathcal{D}(\Phi)$ of decision points. The list $\mathcal{D}$ is initially empty.
The evaluation algorithm Eval that applies the propagation method $(+, \min)$ to the weight-reduced Boolean formula Φ' (and computes $w'(\Phi')$) can be defined as follows. It uses the list of decision points $\mathcal{D}$, and the weight function w as global parameters and α as an oracle. The input parameters are a Boolean function Φ and an adjustment list $\mathcal{A}$ which is needed for adjusting the weight function w.

The verification algorithm $W(X, \alpha)$, given a triple $X = (\Phi, w, t)$ and the certificate α is shown in Alg. 4.3.

5 Analysis

The more close is $\ell = w'(\Phi')$ to the exact value $w(\Phi)$, the more valuable is the result. We show (Theorem 1) that if $\Phi = FG$ and z is the only common variable of F and G, then we can divide the weight $w(z)$ of z between F and

Algorithm 4.1: Algorithm $\mathsf{Desp}(\Phi)$ for generating a list of decision points.

```
Data: Boolean formula $\Phi$
Result: A list $\mathcal{D}$ of decision points
1  if $\Phi = z$ (an atomic variable) then
2      return $\{z\}$;
3  else if $\Phi \in \{FG, F+G\}$
4      $S_F := \mathsf{Desp}(F)$ and $S_G := \mathsf{Desp}(G)$;
5      if $\Phi = FG$ then
6          forall $z \in S_F \cap S_G$ do
7              append $(\Phi, z)$ to $\mathcal{D}$;
8      return $S_F \cup S_G$;
```

Algorithm 4.2: Algorithm $\mathsf{Eval}^\alpha(\Phi, \mathcal{A})$: for evaluating the weight reduced Boolean function Φ'.

```
Data: Boolean formula $\Phi$
Result: Lower bound of $w(\Phi)$
1  if $\Psi = z$ (an atomic variable) then
2      return $w(z) \cdot \prod_{(z,\beta)\in\mathcal{A}} \beta$;
3  if $\Psi = F + G$ then
4      return $\min\{\mathsf{Eval}^\alpha(F,\mathcal{A}), \mathsf{Eval}^\alpha(G,\mathcal{A})\}$;
5  if $\Psi = FG$ then
6      $\mathcal{A}_F := \mathcal{A}_G = \mathcal{A}$;
7      forall $(\Psi, z) \in \mathcal{D}$ do
8          append $(z, \alpha(z))$ to $\mathcal{A}_F$;
9          append $(z, 1-\alpha(z))$ to $\mathcal{A}_G$;
10     return $\mathsf{Eval}^\alpha(\Psi,\mathcal{A}_F) + \mathsf{Eval}^\alpha(\Psi,\mathcal{A}_G)$;
```

G such that the resulting modified function $F'G'$, where z is replaced with z' in F' and with z'' in G' has the same weight than the original function, i.e. $w'(F'G') = w(FG)$. This means that if in all conjunction type sub-formulae FG of Φ the sub-formulae F and G do not have more than one common variable, the certificate α can be chosen so that $\mathsf{Eval}^\alpha(\Phi, \mathcal{A}) = w(\Phi)$, i.e. the lower bound is exact.

Unfortunately, as we show in Theorem 3 that this property does not generalise to the case of more than one common variable. We show that if F and G contain two common variables, then it might be that $\mathsf{Eval}^\alpha(\Phi, \mathcal{A}) < w(\Phi)$ for every possible certificate α, i.e. the *reduction defect* $\delta = w(\Phi) - \max_\alpha \mathsf{Eval}^\alpha(\Phi, \mathcal{A})$ is positive.

Theorem 2 gives necessary and sufficient conditions for the positive reduction defect for a single decision point reduction and shows that the defect a single decision point (Ψ, z) may create is upper bounded by $w(z)/2$.

Algorithm 4.3: Algorithm $W((\Phi, w, t), \alpha)$ for verifying the infeasibility certificate α.

```
Data: Boolean formula $\Phi$
Result: 1 or 0
1 Compute the decision list $\mathcal{D}$ using Desp($\Phi$);
2 Compute $\ell = \mathsf{Eval}(\Phi, ())$, where () is the empty list;
3 if $\ell > t$ then
4     return 1;
5 else
6     return 0;
```

As the weight function w is uniquely defined by its values $w(x_1), w(x_2), \ldots$ on the Boolean variables, we denote by $w_{\xi_1,\xi_2,\ldots}$ the weight function w for which $\xi_1 = w(x_1), \xi_2 = w(x_2), \ldots$. Hence, for any Boolean function Φ, we can define a real-valued function $f_\Phi(\xi_1, \xi_2, \ldots) = w_{\xi_1,\xi_2,\ldots}(\Phi)$. It is easy to see that f_Φ is continuous, because $w(\mu)$ is a continuous function of the weights, the number of min-terms is finite, and for any continuous functions $f(X)$ and $g(X)$, the function $\varphi(X) = \min\{f(X), g(X)\}$ is also continuous.

5.1 The Case of One Common Variable

For convenience, we add additional variables $z, z', z'', \ldots, y_1, y_2, \ldots$ to X. Let $F(x_1, \ldots, x_n, z)$ and $G(z, y_1, \ldots, y_m)$ be monotone Boolean functions. Let z be the only common variable in F and G. Let $F'(x_1, \ldots, x_n, z')$ and $G'(z'', y_1, \ldots, y_m)$ be the modified Boolean functions where z has been replaced with z' and z'', respectively. Note that F' and G' have no common variables. We always assume that $w(z) = w(z') + w(z'')$ even if we change the values of $w(z')$ and $w(z'')$. Thereby, we use a parameter $\alpha \in [0, 1]$ so that $w(z') = \alpha w(z)$ and $w(z'') = (1 - \alpha)w(z)$. By w_α, we mean a weight function that behaves exactly like w, but for which $w(z') = \alpha w(z)$ and $w(z'') = (1 - \alpha)w(z)$. A min-term μ of a Boolean function Φ is said to be α-*cheapest* if $w_\alpha(\mu) = w_\alpha(\Phi)$.

Lemma 1. *$w_\alpha(F') + w_\alpha(G') = w_\alpha(F'G') \leq w(FG)$ for any $\alpha \in [0, 1]$.*

Proof. As F' and G' have no common variables, we have that $w_\alpha(F'G') = w_\alpha(F') + w_\alpha(G')$. For any min-term μ of FG, there is a min-term μ' of $F'G'$ obtained from μ by replacing z with $z'z''$. From $w(z) = w(z') + w(z'')$ it follows that $w_\alpha(\mu') = w_\alpha(\mu)$. Hence, $w_\alpha(F'G') \leq w(FG)$. $\square$

For any conjunction μ, we define two subsets of the unit interval $I = [0, 1]$:

$$\mathcal{O}_{F,\mu} = \{\alpha \in I : w_\alpha(\mu) > w_\alpha(F') \quad \text{and} \quad \mathcal{O}_{G,\mu} = \{\alpha \in I : w_\alpha(\mu) > w_\alpha(G')\}$$

For any Boolean function Φ, let $\mathsf{M}_z(\Phi)$ denote the set of all min-terms of Φ that contain z, and $\mathsf{M}_{\overline{z}}(\Phi)$ the set of all min-terms of Φ without z.

$$\begin{aligned}
\mathcal{O}_F &= \bigcap_{\mu \in \mathsf{M}_{\overline{z}}(F)} \mathcal{O}_{F,\mu} = \{\alpha \in I : \text{All } \alpha\text{-cheapest min-terms of } F' \text{ contain } z'\} \\
\mathcal{O}_G &= \bigcap_{\mu \in \mathsf{M}_{\overline{z}}(G)} \mathcal{O}_{G,\mu} = \{\alpha \in I : \text{All } \alpha\text{-cheapest min-terms of } G' \text{ contain } z''\} \\
\mathcal{U}_F &= \bigcap_{\mu \in \mathsf{M}_{z}(F)} \mathcal{O}_{F,\mu} = \{\alpha \in I : \text{No } \alpha\text{-cheapest min-term of } F' \text{ contains } z'\} \\
\mathcal{U}_G &= \bigcap_{\mu \in \mathsf{M}_{z}(G)} \mathcal{O}_{G,\mu} = \{\alpha \in I : \text{No } \alpha\text{-cheapest min-term of } G' \text{ contains } z''\}
\end{aligned}$$

Lemma 2. *In terms of topology, $\mathcal{O}_F, \mathcal{O}_G, \mathcal{U}_F, \mathcal{U}_G$ ore open subsets of I.*

Proof. As $f_\mu(\alpha) = w_\alpha(\mu) - w_\alpha(F')$ and $g_\mu(\alpha) = w_\alpha(\mu) - w_\alpha(G')$ are continuous functions, $\mathcal{O}_{F,\mu} = f_\mu^{-1}[(0,\infty)]$, and $\mathcal{O}_{G,\mu} = g_\mu^{-1}[(0,\infty)]$, the sets $\mathcal{O}_{F,\mu}$ and $\mathcal{O}_{G,\mu}$ as pre-images of an open set $(0,\infty)$ are open. As finite intersections of open sets, the sets $\mathcal{O}_F, \mathcal{O}_G, \mathcal{U}_F, \mathcal{U}_G$ must also be open. □

Lemma 3. *If $\mathcal{U}_F \cup \mathcal{U}_G \neq I$, then $w_\alpha(F'G') = w(FG)$ for an $\alpha \in I$.*

Proof. Let $\alpha \in I \backslash (\mathcal{U}_F \cup \mathcal{U}_G)$, i.e. there are α-cheapest min-terms μ_F and μ_G of $F'$and G', respectively, such that $\mu_F = x_{i_1}x_{i_2}\ldots x_{i_k}z'$ and $\mu_G = y_{j_1}y_{j_2}\ldots y_{j_m}z''$. Hence, $\mu = x_{i_1}x_{i_2}\ldots x_{i_k}zy_{j_1}y_{j_2}\ldots y_{j_m}$ is a min-term of FG and

$$\begin{aligned}
w_\alpha(F'G') &= w_\alpha(F') + w_\alpha(G') = w_\alpha(\mu_F) + w_\alpha(\mu_G) \\
&= w(x_{i_1}\ldots x_{i_k}) + w_\alpha(z') + w_\alpha(z'') + w(y_{j_1}\ldots y_{j_m}) \\
&= w(x_{i_1}\ldots x_{i_k}) + w(z) + w(y_{j_1}\ldots y_{j_m}) = w(\mu) \geq w(FG) \ ,
\end{aligned}$$

and hence $w_\alpha(F'G') = w(FG)$ by Lemma 1. □

Lemma 4. *If $\mathcal{O}_F \cup \mathcal{O}_G \neq I$, then $w_\alpha(F'G') = w(FG)$ for an $\alpha \in I$.*

Proof. Let $\alpha \in I \backslash (\mathcal{O}_F \cup \mathcal{O}_G)$, i.e. there are α-cheapest min-terms μ_F and μ_G of $F'$and G', respectively, such that $\mu_F = x_{i_1}x_{i_2}\ldots x_{i_k}$ and $\mu_G = y_{j_1}y_{j_2}\ldots y_{j_m}$. Hence, $\mu = x_{i_1}x_{i_2}\ldots x_{i_k}y_{j_1}y_{j_2}\ldots y_{j_m}$ is a min-term of FG and

$$w_\alpha(F'G') = w_\alpha(\mu_F) + w_\alpha(\mu_G) = w(\mu_F) + w(\mu_G) = w(\mu) \geq w(FG) \ ,$$

and hence $w_\alpha(F'G') = w(FG)$ by Lemma 1. □

Lemma 5. *If $\mathcal{O}_F \cup \mathcal{O}_G = I$ and $\mathcal{O}_F$ or $\mathcal{O}_G$ is empty, then $w_\alpha(F'G') = w(FG)$ for an $\alpha \in I$.*

Proof. By $\mathcal{O}_F \cup \mathcal{O}_G = I$, for any $\alpha \in I$, either: (a) all α-cheapest min-terms of F' contain z', or (b) all α-cheapest min-terms of G' contain z''.

If $\mathcal{O}_F = \emptyset$, then this means that for every $\alpha \in I$ there is an α-cheapest min-term of F' without z' contradicting (a), and hence (b) is true. Take $\alpha = 0$ and let $\mu_F = x_{i_1}x_{i_2}\dots x_{i_k}$ be an α-cheapest min-term of F' and μ_G be an α-cheapest min-term of G'. This also means that $\mu = \mu_F\mu_G$ is a min-term of FG, whereas $w(\mu) = w(\mu_F) + w(\mu_G)$. As $w_\alpha(\mu_F) = w(\mu_F)$ and by the choice of α, we have $w_\alpha(\mu_G) = w(\mu_G)$, we have

$$w_\alpha(F'G') = w_\alpha(\mu_F) + w_\alpha(\mu_G) = w(\mu_F) + w(\mu_G) = w(\mu) \geq w(FG) \enspace ,$$

and hence, $w_\alpha(F'G') = w(FG)$ by Lemma 1.

If $\mathcal{O}_G = \emptyset$, then this means that for every $\alpha \in I$ there is an α-cheapest min-term of G' without z'' contradicting (b), and hence (a) is true. Take $\alpha = 1$ and let $\mu_G = y_{j_1}y_{j_2}\dots y_{j_m}$ be an α-cheapest min-term of G' and μ_F be an α-cheapest min-term of F'. This also means that $\mu = \mu_F\mu_G$ is a min-term of FG, whereas $w(\mu) = w(\mu_F) + w(\mu_G)$. As $w_\alpha(\mu_G) = w(\mu_G)$ and by the choice of α, we have $w_\alpha(\mu_F) = w(\mu_F)$, we have

$$w_\alpha(F'G') = w_\alpha(\mu_F) + w_\alpha(\mu_G) = w(\mu_F) + w(\mu_G) = w(\mu) \geq w(FG) \enspace ,$$

and hence, $w_\alpha(F'G') = w(FG)$ by Lemma 1. □

Lemma 6. *If $\mathcal{U}_F \cup \mathcal{U}_G = I$ and $\mathcal{U}_F$ or $\mathcal{U}_G$ is empty, then $w_\alpha(F'G') = w(FG)$ for an $\alpha \in I$.*

Proof. By $\mathcal{U}_F \cup \mathcal{U}_G = I$, for any $\alpha \in I$, either: (a) no α-cheapest min-term of F' contains z', or (b) no α-cheapest min-term of G' contains z''.

If $\mathcal{U}_F = \emptyset$, then for every $\alpha \in I$ there is an α-cheapest min-term of F' with z' contradicting (a), and hence (b) is true. Take $\alpha = 1$ and let $\mu_F = x_{i_1}x_{i_2}\dots x_{i_k}z'$ be an α-cheapest min-term of F' and μ_G be an α-cheapest min-term of G'. Also, $\mu = \mu_F\mu_G$ is a min-term of FG, whereas $w(\mu) = w(\mu_F) + w(\mu_G)$. As $w_\alpha(\mu_G) = w(\mu_G)$ (μ_G does not contain z'') and by the choice of α, we have $w_\alpha(\mu_F) = w(\mu_F)$, we have $w_\alpha(F'G') = w_\alpha(\mu_F) + w_\alpha(\mu_G) = w(\mu_F) + w(\mu_G) = w(\mu) \geq w(FG)$, and hence, $w_\alpha(F'G') = w(FG)$ by Lemma 1.

If $\mathcal{U}_G = \emptyset$, then for every $\alpha \in I$ there is an α-cheapest min-term of G' with z'' contradicting (b), and hence (a) is true. Take $\alpha = 0$ and let $\mu_G = y_{j_1}y_{j_2}\dots y_{j_m}z''$ be an α-cheapest min-term of G' and μ_F be an α-cheapest min-term of F'. This also means that $\mu = \mu_F\mu_G$ is a min-term of FG, whereas $w(\mu) = w(\mu_F)+w(\mu_G)$. As $w_\alpha(\mu_F) = w(\mu_F)$ (μ_F does not contain z') and by the choice of α, we have $w_\alpha(\mu_G) = w(\mu_G)$, we have $w_\alpha(F'G') = w_\alpha(\mu_F) + w_\alpha(\mu_G) = w(\mu_F) + w(\mu_G) = w(\mu) \geq w(FG)$, and hence, $w_\alpha(F'G') = w(FG)$ by Lemma 1. □

Theorem 2. *There is $\alpha \in I$, so that $w_\alpha(F'G') = w(FG)$.*

Proof. Assume on the contrary that the statement is false. Then by the lemmas above, $\mathcal{O}_F \cup \mathcal{O}_G = I$, $\mathcal{U}_F \cup \mathcal{U}_G = I$ and all $\mathcal{O}_F, \mathcal{O}_G, \mathcal{U}_F, \mathcal{U}_G$ are non-empty. As I is connected, we have $\mathcal{O}_F \cap \mathcal{O}_G \neq \emptyset$ and $\mathcal{U}_F \cap \mathcal{U}_G \neq \emptyset$.

If there is a cheapest min-term of FG that contains z and $\mu = x_{i_1}x_{i_2}\dots x_{i_k}$ $zy_{j_1}y_{j_2}\dots y_{j_m}$ is such a min-term, and $\alpha \in \mathcal{U}_F \cap \mathcal{U}_G$, then no α-cheapest min-terms of F' and G' contain z' or z'' and hence $w_\alpha(F') = w(F)$ and $w_\alpha(G) = w(G)$. As $\mu_F = x_{i_1}x_{i_2}\dots x_{i_k}z'$ and $\mu_G = z''y_{j_1}y_{j_2}\dots y_{j_m}$ are min-terms of F'

and G', respectively (but not α-cheapest, as they contain z' and z''), then we have a contradiction:

$$w(FG) = w_\alpha(\mu_F) + w_\alpha(\mu_G) > w_\alpha(F') + w_\alpha(G') = w(F) + w(G) \geq w(FG) \ .$$

If no cheapest min-term of FG contains z, and $\alpha \in \mathcal{O}_F \cap \mathcal{O}_G$, then all α-cheapest min-terms of F' and G' contain z' or z''. Let $\mu_F = x_{i_1}x_{i_2}\dots x_{i_k}z'$ and $\mu_G = z''y_{j_1}y_{j_2}\dots y_{j_m}$ be α-cheapest min-terms of F' and G', respectively. As then $\mu = x_{i_1}x_{i_2}\dots x_{i_k}zy_{j_1}y_{j_2}\dots y_{j_m}$ is a min-term of FG (but not the cheapest, as it contains z), we again have a contradiction: $w_\alpha(F'G') = w_\alpha(\mu_F) + w_\alpha(\mu_G) = w(\mu) > w(FG) \geq w_\alpha(F'G')$. □

5.2 Two or More Common Variables

Let z be any common variable of F and G, such that $F = F_0 + z\partial F$ and $G = G_0 + z\partial G$, where $\partial F, \partial G, F_0, G_0$ do not depend on z. The functions ∂F and ∂G are the so-called *Boolean derivatives* $\partial F = \frac{\partial}{\partial z}F$ and $\partial G = \frac{\partial}{\partial z}G$. Let $F' = F_0 + z'\partial F$ and $G' = G_0 + z''\partial G$, where $w(z') + w(z'') = w(z)$; $w(z') = \alpha w(z)$ and $w(z'') = (1-\alpha)w(z)$, where $\alpha \in [0,1]$; and $m_0 = \min\{w(F_0G_0), w(z\partial F\partial G)\}$. As

$$\begin{aligned} FG &= (F_0 + z\partial F)(G_0 + z\partial G) = F_0G_0 + z\partial F\partial G + zG_0\partial F + zF_0\partial G \\ F'G' &= (F_0 + z'\partial F)(G_0 + z''\partial G) = F_0G_0 + z'z''\partial F\partial G + z'G_0\partial F + z''F_0\partial G \end{aligned}$$

and $w(z'z''\partial F\partial G) = w(z\partial F\partial G)$, we have

$$\begin{aligned} w(FG) &= \min\{w(F_0G_0), w(z\partial F\partial G), w(zG_0\partial F), w(zF_0\partial G)\} \\ &= \min\{m_0, w(zG_0\partial F), w(zF_0\partial G)\} \\ w(F'G') &= \min\{w(F_0G_0), w(z\partial F\partial G), w(z'G_0\partial F), w(z''F_0\partial G)\} \\ &= \min\{m_0, w(z'G_0\partial F), w(z''F_0\partial G)\} \ . \end{aligned}$$

The difference $\delta = w(FG) - \max_\alpha w(F'G')$ is called the *reduction defect*. It is non-negative, as $w(FG) \geq w(F'G')$.

Theorem 3. *The reduction defect δ is non-zero ($\delta > 0$) if and only if the following inequalities hold:*

$$\mid w(G_0\partial F) - w(F_0\partial G) \mid < w(z) < 2m_0 - w(G_0\partial F) - w(F_0\partial G) \ , \qquad (1)$$

and has upper bound $\delta \leq \frac{w(z)}{2} - \frac{1}{2} \mid w(G_0\partial F) - w(F_0\partial G) \mid \leq \frac{w(z)}{2}$.

Proof. Note that δ is non-zero if and only if for any α:

$$\begin{aligned} m' &= \min\{w(z'G_0\partial F), w(z''F_0\partial G)\} < \min\{w(zG_0\partial F), w(zF_0\partial G)\} \qquad (2) \\ m' &< m_0 = \min\{w(F_0G_0), w(z\partial F\partial G)\} \qquad (3) \end{aligned}$$

The quantity $m_1(\alpha) = \min\{w(z'G_0\partial F), w(z''F_0\partial G)\}$ gains maximum if $w(z'G_0\partial F) = w(z''F_0\partial G)$. We study two cases: (a) $w(G_0\partial F) \geq w(F_0\partial G)$ and (b) $w(G_0\partial F) \leq w(F_0\partial G)$.

From (a), it follows that the maximum occurs if

$$w(z) - 2w(z') = w(z'') - w(z') = w(G_0\partial F) - w(F_0\partial G) \ ,$$

i.e. if $w(z') = w_0 = \frac{w(z)}{2} - \frac{1}{2}\left(w(G_0\partial F) - w(F_0\partial G)\right)$. Hence,

$$\begin{aligned} M &= \max_\alpha m_1(\alpha) = \max_\alpha \min\{w(z'G_0\partial F), w(z''F_0\partial G)\} = w_0 + w(G_0\partial F) \\ &= \frac{w(z)}{2} - \frac{1}{2}\left(w(G_0\partial F) - w(F_0\partial G)\right) + w(G_0\partial F) \\ &= \frac{w(z)}{2} + \frac{1}{2}\left(w(G_0\partial F) + w(F_0\partial G)\right) \ . \end{aligned}$$

Considering that $\min\{w(zG_0\partial F), w(zF_0\partial G)\} = w(z) + w(F_0\partial G)$ and $\mid w(G_0\partial F) - w(F_0\partial G) \mid = w(G_0\partial F) - w(F_0\partial G)$ by (a), the conditions (2) and (3) transform to

$$\begin{aligned} &\frac{w(z)}{2} + \frac{1}{2}\left(w(G_0\partial F) + w(F_0\partial G)\right) < w(z) + w(F_0\partial G) \\ &\frac{w(z)}{2} + \frac{1}{2}\left(w(G_0\partial F) + w(F_0\partial G)\right) < m_0 \ , \end{aligned}$$

which is equivalent to (1). The reduction defect is:

$$\begin{aligned} \delta &= w(FG) - \max_\alpha w(F'G') = \min\{m_0, w(zG_0\partial F), w(zF_0\partial G)\} - M \\ &\leq \min\{w(zG_0\partial F), w(zF_0\partial G)\} - M \\ &= w(z) + w(F_0\partial G) - \frac{w(z)}{2} - \frac{w(G_0\partial F) + w(F_0\partial G)}{2} \\ &= \frac{w(z)}{2} - \frac{1}{2}\left(w(G_0\partial F) - w(F_0\partial G)\right) = \frac{w(z)}{2} - \frac{1}{2} \mid w(G_0\partial F) - w(F_0\partial G) \mid \ . \end{aligned}$$

From (b), it follows that the maximum occurs if

$$w(z) - 2w(z'') = w(z') - w(z'') = w(F_0\partial G) - w(G_0\partial F) \ ,$$

i.e. if $w(z'') = w_0' = \frac{w(z)}{2} - \frac{1}{2}\left(w(F_0\partial G) - w(G_0\partial F)\right)$. Hence,

$$\begin{aligned} M &= \max_\alpha m_1(\alpha) = \max_\alpha \min\{w(z'G_0\partial F), w(z''F_0\partial G)\} = w_0' + w(F_0\partial G) \\ &= \frac{w(z)}{2} - \frac{1}{2}\left(w(F_0\partial G) - w(G_0\partial F)\right) + w(F_0\partial G) \\ &= \frac{w(z)}{2} + \frac{1}{2}\left(w(F_0\partial G) + w(G_0\partial F)\right) \ . \end{aligned}$$

As $\min\{w(zG_0\partial F), w(zF_0\partial G)\} = w(z) + w(G_0\partial F)$ and $\mid w(G_0\partial F) - w(F_0\partial G) \mid = w(F_0\partial G) - w(G_0\partial F)$ by (b), the conditions (2) and (3) transform to

$$\begin{aligned} &\frac{w(z)}{2} + \frac{1}{2}\left(w(G_0\partial F) + w(F_0\partial G)\right) < w(z) + w(G_0\partial F) \\ &\frac{w(z)}{2} + \frac{1}{2}\left(w(G_0\partial F) + w(F_0\partial G)\right) < m_0 \ , \end{aligned}$$

which is equivalent to (1). The reduction defect is:

$$\begin{aligned}\delta &\le w(z) + w(G_0\partial F) - \frac{w(z)}{2} - \frac{w(G_0\partial F) + w(F_0\partial G)}{2} \\ &= \frac{w(z)}{2} - \frac{1}{2}\left(w(F_0\partial G) - w(G_0\partial F)\right) \\ &= \frac{w(z)}{2} - \frac{1}{2}\mid w(F_0\partial G) - w(G_0\partial F) \mid \quad . \qquad \square\end{aligned}$$

Theorem 4. *There are F and G with two common variables and a suitable distribution weights, so that the weight reduction always has a defect.*

Proof. Let $F = vx + auz$ and $G = uy + bz$, i.e. the common variables are z and u. Assume that all weights are equal to 1. Hence, $w(FG) = w(xyvu + xvbz + ayuz + abuz) = \min\{w(xyvu), w(xvbz), w(ayuz), w(abuz)\} = 4$. If $w(u') + w(u'') = w(u)$ and $w(z') + w(z'') = w(z)$, we have $F'G' = (vx + au'z')(u''y + bz'') = vxyu'' + xvbz'' + ayu'u''z' + abuz'z''$. Hence,

$$\begin{aligned}w(F'G') &= \min\{w(vxyu''), w(xvbz''), w(ayu'u''z'), w(abuz'z'')\} \\ &\le \min\{w(xvbz''), w(ayu'u''z')\} = \min\{w(xvbz''), w(ayuz')\} \\ &= \min\{w(xvb) + w(z''), w(ayu) + w(z')\} \\ &= 3 + \min\{w(z'), w(z'')\} \le 3.5 \quad . \qquad \square\end{aligned}$$

6 Conclusions and Open Questions

The results show that the new method is applicable in practice, as real-life attack trees tend to be flat with relatively small number of decision points, which makes it possible to generate infeasibility certificates for such trees. Even the relative error is close to 0.5 may be acceptable in practice because the attack tree parameters often cannot be measured with high precision.

Theorem 2 may be pessimistic in its assessment because it considers a particular way of choosing the certificate. For any decision point (Ψ, z) it tries to choose $\alpha(\Psi, z)$ so that it minimizes the defect of a single step of the reduction. Global methods of choosing the values of α may obtain better results than what can be achieved by minimizing the local reduction defects. More precise estimations of reduction defect would be in place.

References

1. A. A. Ahmadi, A. Olshevsky, P. A. Parrilo, and J. N. Tsitsiklis. NP-hardness of deciding convexity of quartic polynomials and related problems. *Mathematical Programming*, 137(1):453–476, 2013.
2. A. Buldas, P. Laud, J. Priisalu, M. Saarepera, and J. Willemson. Rational choice of security measures via multi-parameter attack trees. In J. Lòpez, editor, *CRITIS*, volume 4347 of *Lecture Notes in Computer Science*, pages 235–248. Springer, 2006.

3. A. Buldas and A. Lenin. New efficient utility upper bounds for the fully adaptive model of attack trees. In S. K. Das, C. Nita-Rotaru, and M. Kantarcioglu, editors, *GameSec*, volume 8252 of *Lecture Notes in Computer Science*, pages 192–205. Springer, 2013.
4. A. Buldas and R. Stepanenko. Upper bounds for adversaries' utility in attack trees. In J. Grossklags and J. C. Walrand, editors, *GameSec*, volume 7638 of *Lecture Notes in Computer Science*, pages 98–117. Springer, 2012.
5. G. Blekherman, P. A. Parrilo, R. R. Thomas, G. Blekherman, P. A. Parrilo, and R. R. Thomas. *Semidefinite Optimization and Convex Algebraic Geometry.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2012.
6. S.Boyd and L.Vandenberghe. Convex Optimization. Cambridge University Press, New York, NY, USA, 2004.
7. P. Corbineau. *A Declarative Language for the Coq Proof Assistant*, pages 69–84. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
8. J. A. De Loera, J. Lee, P. N. Malkin, and S. Margulies. Computing infeasibility certificates for combinatorial problems through hilbert's nullstellensatz. *J. Symb. Comput.*, 46(11):1260–1283, Nov. 2011.
9. J. W. Helton and J. Nie. Semidefinite representation of convex sets. *Mathematical Programming*, 122(1):21–64, 2010.
10. C. J. Hillar and L.-H. Lim. Most tensor problems are np-hard. *J. ACM*, 60(6):45:1–45:39, Nov. 2013.
11. A. Jürgenson and J. Willemson. Computing exact outcomes of multi-parameter attack trees. In R. Meersman and Z. Tari, editors, *OTM Conferences (2)*, volume 5332 of *Lecture Notes in Computer Science*, pages 1036–1051. Springer, 2008.
12. A. Jürgenson and J. Willemson. Serial model for attack tree computations. In D. Lee and S. Hong, editors, *ICISC*, volume 5984 of *Lecture Notes in Computer Science*, pages 118–128. Springer, 2009.
13. A. Jürgenson and J. Willemson. On fast and approximate attack tree computations. In J. Kwak, R. H. Deng, Y. Won, and G. Wang, editors, *ISPEC*, volume 6047 of *Lecture Notes in Computer Science*, pages 56–66. Springer, 2010.
14. A. Lenin. *Reliable and Efficient Determination of the Likelihood of Rational Attacks.* TUT Press, 2015.
15. S. Mauw and M. Oostdijk. Foundations of attack trees. In D. Won and S. Kim, editors, *ICISC*, volume 3935 of *Lecture Notes in Computer Science*, pages 186–198. Springer, 2005.
16. L. de Moura and N. Bjørner. *Z3: An Efficient SMT Solver*, pages 337–340. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
17. S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo. Sostools: Sum of squares optimization toolbox for matlab, 2004.
18. B. Schneier. Attack trees. *Dr. Dobb's Journal of Software Tools*, 24(12):21–22, 24, 26, 28–29, Dec. 1999.
19. K. E. Smith, L. Kahanpää, and P. K. [et al.]. *An invitation to algebraic geometry.* Universitext. Springer Science + Business Media, 2004, New York, 2000.
20. G. Stengle. A nullstellensatz and a positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207:87–98, 1974.
21. G. Stengle. A nullstellensatz and positivstellensatz in semialgebraic geometry. *Math. Ann.*, 207:87–97, 1994.