A Unified Compositional View of Attack Tree Metrics

Benedikt Peterseim and Milan Lopuhaä-Zwakenberg,
University of Twente

***Abstract*—Attack trees (ATs) are popular graphical models for reasoning about the security of complex systems, allowing for the quantification of risk through so-called AT metrics. A large variety of different such AT metrics have been proposed, and despite their wide-spread practical use, no systematic treatment of attack tree metrics so far is fully satisfactory. Existing approaches either fail to include important metrics, or they are too general to provide a useful systematic way for defining concrete AT metrics, giving only an abstract characterisation of their behaviour. We solve this problem by developing a compositional theory of ATs and their functorial semantics based on gs-monoidal categories. Viewing attack trees as string diagrams, we show that components of ATs form a channel category, a particular type of gs-monoidal category. AT metrics then correspond to functors of channel categories. This characterisation is both general enough to include all common AT metrics, and concrete enough to define AT metrics by their logical structure.**

I. Introduction

Background: Attack trees. Since their inception [Sch99], attack trees (ATs) have been widely applied across numerous domains, including nuclear control systems [KS07], smart grids [BHK+14], and railway control systems [DWT17]. How they work is best illustrated by an example: Figure 1 shows a simple example of an AT. Note that an AT is not necessarily a tree in the graph-theoretic sense: some nodes (such as `forge badge` in Figure 1) may have multiple parents. For a comprehensive overview of ATs in comparison to other graphical models for security modelling, see [KPCS14, Section 3.1.1]. A number of extensions to the formalism of ATs have been proposed, such as dynamic ATs [JKM+15] and attack-defence trees [KMRS10]. While we focus on standard ATs for simplicity, our methods will be set up in a way that can easily be adapted to these extensions.

Attack tree metrics. In addition to a *qualitative* model of how basic attack steps combine via AND and OR gates, ATs allow for the quantification of risk through AT metrics. For example, in Figure 1, we use the *min. cost metric*, associating to each basic attack step the minimum cost required to successfully perform it. The question of computing the *metric value* is now: how do these attribute values propagate from the basic attack steps to the top node? In our example: what is the minimum cost required to successfully gain access to the office space?

Already in the example of the min. cost metric, this question may be answered in different ways: depending on how one interprets the AT, the minimum cost of a successful attack may either be \$110 or \$100 for the AT in Figure 1; see Section II-A. In addition, many other types of metrics are used, including the minimum time or skill level required for a successful attack, or its maximal probability (under certain assumptions on attacker behaviour). With each of the various ways to quantify risk in an AT admitting several interpretations as an AT metric, the need for a systematic unified account of AT metrics emerges:

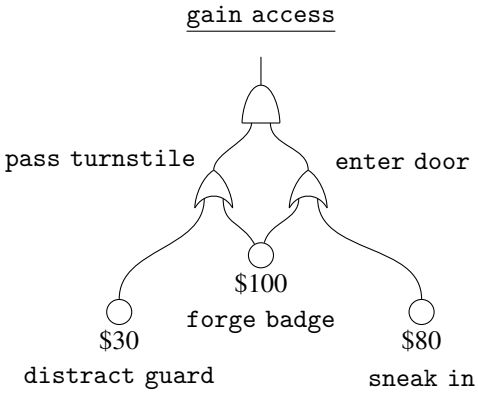


Fig. 1. *Example attack tree:* An attacker aims to gain unauthorised access to an office space. This requires passing a turnstile and entering through a secured door (top AND gate). To pass the turnstile, they can either forge an access badge or distract the guard and jump over (left OR gate). To enter the door, they can either also use a forged badge or follow an employee in (right OR gate). Basic attack steps are labelled with their cost: distracting the guard costs \$30, forging a badge \$100, and sneaking in \$80.

> **Problem I.1.** Provide a framework for attack tree metrics that is both:
> 1) sufficiently *general* to capture all common attack tree metrics used in practice, and
> 2) sufficiently *concrete* for attack tree metrics to be fully determined by how they behave on the basic building blocks of attack trees: logic gates and basic attack steps.

The second property ensures that a framework for AT metrics yields a systematic method for defining AT metrics, instead of just an abstract characterisation of their general behaviour.

Shortcomings of existing frameworks. Despite the widespread use of ATs, so far, no systematic treatment of AT metrics exists that meets both of the requirements of Theorem I.1.

Several mutually incompatible frameworks based on *semirings* (see Theorem V.1) exist, one based on what we will refer to as the *bottom-up interpretation* [MO05], one based on the *propositional interpretation* [LZBS22], and another one

based on yet a different interpretation [BK17]. All are concrete enough to provide a recipe for defining AT metrics as required by our second desideratum. However, they do not satisfy our first requirement, failing to capture those AT metrics that do not adhere to their chosen interpretation.

In the other direction, earlier work based on *operads* [LZ24] provided a framework that does include essentially every important AT metric, instead failing to satisfy our second requirement of sufficient concreteness. In particular, specifying an AT metric as defined in this framework requires specifying its value on infinitely many ATs, which is clearly not satisfactory.

Approach. We introduce a new framework for AT metrics that solves Problem I.1. The key idea is to consider building blocks of ATs with multiple inputs and outputs. We define a metric by the operations it associates to AND/OR gates, and how it handles the dependence between multiple outputs. This is general enough to capture all relevant metrics, and concrete enough to specify metrics from finite data. We differ from semiring formalisms (which only describe the AND/OR operators and hardcode the dependency handling) and from the operad approach (which needs to specify infinitely many operators, so it cannot be used computationally in full generality).

Mathematically, our framework is based on *gs-monoidal categories*, a category-theoretic concept originally proposed in the context of term rewriting [Gad96]. Nevertheless, we will assume no prior knowledge of category theory, as we give a self-contained definition of a special type of gs-monoidal category we call *channel categories* for their resemblance to information-theoretic channels. A channel category is defined in terms of abstract operations of parallel and sequential composition: how a metric handles these operations exactly describes how a metric handles dependencies within ATs. We show that attack trees give rise to a channel category, and that the same holds for most metrics from the literature (with only a few exceptions, discussed in Section V-F). Thus, we define an AT metric as a *functor of channel categories*. Despite its generality, this definition is still sufficiently concrete to yield a method for computing the metric. Interestingly, the different AT semantics that have been proposed in the literature [MO05], [KPCS14], [LZBS22] can also be understood as functors of channel categories; hence our approach unifies AT metrics and semantics.

Contributions. In summary, our main contributions are:

- A new, compositional framework for AT metrics;
- A self-contained exposition of the underlying mathematical theory of channel categories;
- Showing most existing metrics fit into this framework;
- Unifying AT metrics and semantics.

Understanding the compositionality of ATs is also of independent interest, beyond the problem of systematising AT metrics.

First, it provides a more formal view on how modelling with ATs is done *compositionally*, by building up models of complex systems from simpler ones. In the context of *fault trees* (the counterpart of ATs in reliability engineering), *component fault trees* already provide such formalism for compositional modelling [KLM03]. Our formalism is more general, since fault trees can be viewed as a special case of ATs in our setting (where the "attacker" is nature).

Moreover, our treatment of ATs as *term graphs* (see Section III) presents them as particular kinds of *string diagrams*, fitting into a wider picture. String diagrams have broadly been applied for a compositional understanding of graphical languages across various domains: Bayesian networks [Fon13], [Jac18], quantum circuits [CK18], and tensor networks [BCJ11]. "String diagram" is an umbrella term used for various related graphical calculi, and term graphs are the specific form of string diagram corresponding to the context of channel categories.

Finally, compositional semantics are an inherently desirable property of any formal language, ensuring that the meaning of complex terms is systematically built up from the meanings of simpler terms. Our results can then also be interpreted as providing **functorial (or "categorical") semantics** for ATs.

II. Example: Two Min. Cost Metrics

In this section, we highlight the key aspects of our approach through a concrete example. We show how two seemingly incompatible interpretations of the min. cost metric, the *bottom-up* [MO05] and the *propositional* [LZBS22] interpretation, can be defined in a unified way within our framework.

A. Two interpretations

The bottom-up interpretation [MO05]. In this interpretation, the AND and OR gates directly denote operations that combine metric values. In the case of minimum cost, an AND gate is interpreted as taking the sum of the costs of its children, since *all* sub-goals must be achieved to successfully execute the attack associated to the AND gate. OR gates are interpreted as taking the minimum of the costs of its children, as only *some* sub-goal needs to be achieved for the OR gate to be activated. Hence, in the example AT from Figure 1, the minimum cost of a successful attack under this bottom-up interpretation is:

$$\min(\$30, \$100) + \min(\$100, \$80) = \$110.$$

Importantly, under the bottom-up interpretation, the basic attack step labelled `forge badge` needs to be executed separately in each sub-goal in which it appears. That is, the attacker cannot use the same access badge for both passing the turnstile and for entering through the office door. If instead we would like to model the same forged access badge being used in both sub-goals, we either need a different attack tree, or we may use the *propositional* interpretation.

The propositional interpretation [LZBS22]. This interpretation views an attack tree as a propositional formula, describing which sets of basic attack steps lead to achieving the top-level goal. Such set of basic attack steps is called a *successful attack*. The minimum cost is computed by considering all *minimal* successful attacks, those successful attacks that have

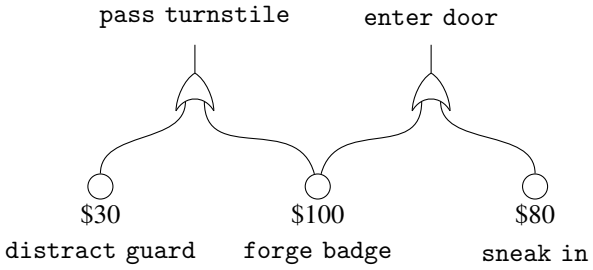
no proper successful subsets. For the example attack tree from Figure 1, there are two minimal successful attacks: one where the attacker performs only the basic attack step labelled `forge badge`; and one attack where the steps labelled `distract guard` and `sneak in` (but not `forge badge`) are performed. These minimal successful attacks have costs \$100 and \$30 + \$80, respectively. Therefore, the minimum cost is

$$\min(\$100, \$30 + \$80) = \$100.$$

Since the attacker can use the same forged access badge for both passing the turnstile and entering through the office door, the minimum cost of a successful attack is strictly lower than in the bottom-up interpretation. In particular, the two interpretations are not equivalent.

B. A unified way to define both attack tree metrics

The reason why these two interpretations of min. cost yield different values lies in how they treat *dependence*. To make this more mathematically precise, consider the "multiple-root attack tree" resulting from removing the top-level AND-gate of the example attack tree in Figure 1. The two top-level goals in this example are `pass turnstile` and `enter door`:



(1)

In the bottom-up interpretation, evaluating a multiple-root attack tree is straightforward: we compute the cost of each output independently, giving output vector $\begin{pmatrix}80 & 30\end{pmatrix}^\intercal$ (reading outputs from left to right). Taking their sum, the metric value of the larger AT can be computed from these values alone.

In contrast, simply storing the outputs $\begin{pmatrix}80 & 30\end{pmatrix}^\intercal$ is not enough for the propositional interpretation, as it does not tell us that we can activate both top gates more cheaply by the single basic attack step labelled `forge badge`. To account for this, our output is a four-dimensional vector, whose entries correspond to the minimum cost required to obtain top-level outputs 00, 01, 10, 11; the result is $\begin{pmatrix}0 & 80 & 30 & 100\end{pmatrix}^\intercal$. This is all the information later gates need to compute min. costs according to the propositional interpretation. AND/OR gates act on such vectors as matrices over the $(\min, +)$-semiring. In fact, the top AND gate becomes the matrix $\left(\begin{smallmatrix}0 & 0 & 0 & \infty\\ \infty & \infty & \infty & 0\end{smallmatrix}\right)$, so the result is:

$$\begin{pmatrix}0 & 0 & 0 & \infty\\ \infty & \infty & \infty & 0\end{pmatrix}\begin{pmatrix}0\\80\\30\\100\end{pmatrix} = \begin{pmatrix}\min(0+0, 0+80, 0+30, \infty+100)\\ \min(\infty+0, \infty+80, \infty+30, 0+100)\end{pmatrix} = \begin{pmatrix}0\\100\end{pmatrix}.$$

The first coefficient is the minimum cost to get output 0 for the overall AT, and the second coefficient the minimal cost to get output 1; this is the \$100 we computed before.

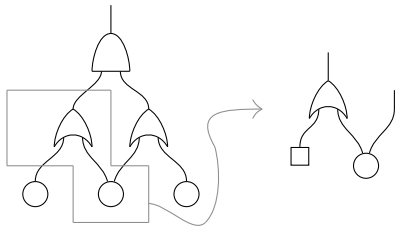
This perspective allows us to formalise both interpretations as mappings of the form:

$$[\![\cdot]\!]_{\mathsf{bottom\text{-}up}} : \mathsf{AttackTrees} \longrightarrow \mathsf{Functions},$$
$$[\![\cdot]\!]_{\mathsf{propositional}} : \mathsf{AttackTrees} \longrightarrow \mathsf{Matrices},$$

where AttackTrees denotes a certain kind of category of attack trees, and Functions and Matrices are categories of the same kind capturing the semantics of the bottom-up and propositional interpretations, respectively.

More precisely, these mappings are *functors of channel categories*. Informally, a channel category consists of sets of "channels" (such as functions or matrices), together with operations for sequential and parallel composition of these channels, and a number of special "wiring" channels.

To understand attack trees in a compatible way, we need to pass to *attack tree components*. Attack tree components are parts of an attack tree, cut out from T along edges:



(2)

In contrast to a full attack tree, an attack tree component may have multiple inputs, denoted by small squares ("□"), and outputs, represented by outgoing wires. For example, the attack tree component on the right hand side of (2) has one input and two outputs; the multiple-root attack tree (1) has two outputs and no input. Section III will make attack tree components precise, by defining them as certain *term graphs*. Channel categories will be introduced in detail in Section IV.

The perspective of attack tree metrics as functors of channel categories accounts not only for the two instantiations of the min. cost metric discussed here, but also for a wide range of metrics and semantics found in the literature. This is the subject of Section V.

III. Syntax: Term Graphs

On a syntactic level, graphical risk models such as attack trees, fault trees and their numerous variants can all be understood as *term graphs*. Term graphs can be viewed as the single-sorted ("untyped") special case of *gs-monoidal string diagrams* [CG99], [FL23], [CJ19], but we will not assume any familiarity with string diagrams or category theory. Alternatively, term graphs can be viewed as a generalisation of syntax trees, allowing for explicit sharing between subtrees.

Our main focus will be on particular term graphs: attack tree components. Nevertheless, we work in the general setting of term graphs for two reasons. First, it allows us to directly apply the results of [CG99] to our case of interest. Second, it

provides a basis to easily generalise our results to more general variants of attack trees.

Term graphs are always defined with respect to a certain *signature*.

Definition III.1 (Signature)**.** A (single-sorted, algebraic) *signature* is a set Σ together with a function $\mathsf{arity}_\Sigma : \Sigma \to \mathbb{N}$. The elements of Σ are called *function symbols* and each $f \in \Sigma$ is thought of as taking $\mathsf{arity}_\Sigma(f)$ many arguments.

Example III.2. A typical example of a signature in the context of algebra is the signature of groups,

$$\mathsf{Gr} := \{m, i, 1\},$$
$$\mathsf{arity}_{\mathsf{Gr}}(m) = 2,\ \mathsf{arity}_{\mathsf{Gr}}(i) = 1,\ \mathsf{arity}_{\mathsf{Gr}}(1) = 0.$$

Here, m denotes the group operation, or multiplication (a binary operation); i the inversion (a unary operation); and 1 the identity element (a nullary operation, or constant).

Our signature of interest is the following.

Definition III.3. Let B be a set, thought of as consisting of *basic attack step labels*. The *signature of attack trees over* B is

$$\mathsf{AT}(B) := \{\mathtt{AND}_i, \mathtt{OR}_i \mid i \in \mathbb{N}_{\geq 1}\} \sqcup B,$$
$$\mathsf{arity}_{\mathsf{AT}(B)}(\mathtt{AND}_i) = \mathsf{arity}_{\mathsf{AT}(B)}(\mathtt{OR}_i) = i,$$
$$\mathsf{arity}_{\mathsf{AT}(B)}(b) = 0,$$

for all $i \in \mathbb{N}_{\geq 1}, b \in B$. Here, $\sqcup$ denotes the disjoint union of sets.

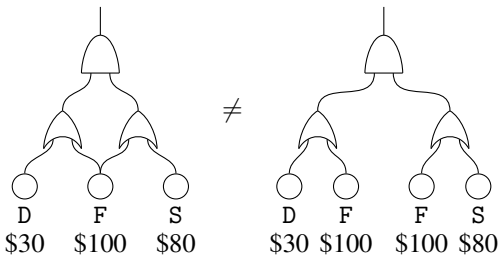
For each $i \in \mathbb{N}_{\geq 1}$, the function symbols $\mathtt{AND}_i$ and $\mathtt{OR}_i$ are thought of as representing an AND gate and an OR gate with i inputs, respectively.

Example III.4. The AT from Figure 1 is defined over the signature $\mathsf{AT}(B)$ with basic attack steps labels

$$B := \{\texttt{distract guard}, \texttt{forge badge}, \texttt{sneak in}\}.$$

We will abbreviate $B = \{\mathtt{D}, \mathtt{F}, \mathtt{S}\}$ from now on.

At this point, one might be tempted to expect that term graphs are simply graph representations of terms over the given signature. This is *not* the case, as term graphs allow for explicit sharing of information. For example, as a consequence of the discussion in Section II, we see that the following two attack trees are different under the propositional interpretation:



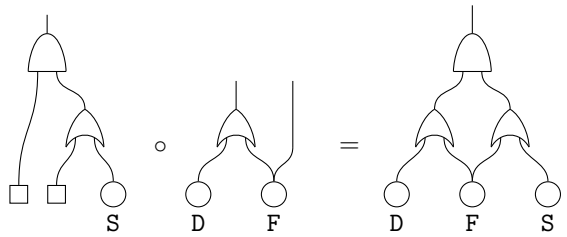
(3)

On the left-hand-side tree, the minimum cost of a successful attack is \$100, whereas on the right-hand-side tree, it is \$110. However, if we were to view these two attack trees as ordinary syntax graphs, where one freely merges isomorphic subgraphs, they would both represent the same term:

$$\mathtt{AND}_2(\mathtt{OR}_2(\mathtt{D}, \mathtt{F}), \mathtt{OR}_2(\mathtt{F}, \mathtt{S})).$$

The ability to explicitly treat sharing is the first key difference between ordinary, algebraic terms and term graphs.

The second key difference is that term graphs, in the sense of Theorem III.5 below, allow for multiple *inputs* and *outputs*. This enables us to compose term graphs such as attack trees from smaller ones, generalising the *modular composition* of attack trees [LZ24, Definition II.4]. For example:



(4)

Here, input nodes are represented by small squares ("□") and output nodes are the ends of wires not connected to any other node. Both are ordered and read left to right.

We now introduce term graphs and their compositional structure in detail, following the general theory laid out in [CG99] for everything not specific to attack trees. In the following definition, we use the notation $\mathsf{List}(X)$ for the set of lists (i.e. finite, ordered sequences) over a set X, and $\mathsf{set}(l)$ for the set of all elements appearing in a list l.

Definition III.5 (Term graph)**.** A *term graph* T over a signature Σ, or Σ*-term graph*, is a tuple

$$T = (N^T, \mathsf{in}^T, \mathsf{out}^T, \mathsf{label}^T, \mathsf{ch}^T),$$

where:

1) N^T is a finite set of *nodes*.
2) $\mathsf{in}^T \in \mathsf{List}(N^T)$ is a list of *input nodes*, and
3) $\mathsf{out}^T \in \mathsf{List}(N^T)$ is a list of *output nodes*.
4) $\mathsf{label}^T \colon N^T \setminus \mathsf{set}(\mathsf{in}^T) \to \Sigma$ is a function assigning to each non-input node a function symbol from Σ.
5) $\mathsf{ch}^T \colon N^T \setminus \mathsf{set}(\mathsf{in}^T) \to \mathsf{List}(N)$ is a function assigning to each non-input node n its list of *children* $\mathsf{ch}^T(n)$,

These data are required to satisfy the following axioms.

1) For each non-input node $n \in N^T$,
$$\mathsf{arity}_\Sigma(\mathsf{label}^T(n)) = \mathrm{length}(\mathsf{ch}^T(n)).$$
2) The directed graph (N^T, E^T) is acyclic, where
$$E^T := \{(n, m) \in N^T \times N^T \mid m \in \mathsf{ch}^T(n)\}.$$
3) The list of input nodes in^T contains no duplicates.

We write $T \colon i \to j$ for term graphs with i inputs and j outputs. An *isomorphism of term graphs* S and T is a bijection $N^S \to N^T$ preserving the input and output nodes (including their ordering), the labelling function, and the child function.

We will generally consider term graphs up to isomorphism, treating isomorphic term graphs as equal.

The set of all Σ-term graphs with i inputs and j outputs (up to isomorphism) is denoted $\mathsf{TermGraphs}_\Sigma(i,j)$.

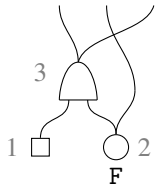
From this perspective, attack trees are term graphs with 0 inputs and 1 output:

Definition III.6 (Attack tree)**.** Let B be a set (thought of as consisting of basic attack step labels). An *attack tree* T over B is an $\mathsf{AT}(B)$-term graph $T\colon 0 \to 1$.

We may think of general $\mathsf{AT}(B)$-term graphs as representing *components* of attack trees, yet to be composed into a full attack tree. We will therefore use the following terminology.

Definition III.7 (Attack tree component)**.** Let B be a set. An *attack tree component* over B is an $\mathsf{AT}(B)$-term graph.

Example III.8. Consider the following attack tree component over $B = \{\mathtt{D}, \mathtt{F}, \mathtt{S}\}$:



It can be defined more precisely as a term graph $T\colon 1 \to 3$ with one input and three outputs, and the following data:

1) Three nodes, arbitrarily chosen to be: $N^T := \{1,2,3\}$,
2) A single-element list of input nodes: $\mathsf{in}^T := [1]$,
3) A three-element list of output nodes: $\mathsf{out}^T := [3,2,3]$,
4) A labelling function $\mathsf{label}^T\colon N^T \setminus \mathsf{set}\big(\mathsf{in}^T\big) \to \mathsf{AT}(B)$ given by $\mathsf{label}^T(2) = \mathtt{F}, \mathsf{label}^T(3) = \mathtt{AND}_2$.
5) A child function $\mathsf{ch}^T\colon N^T \setminus \mathsf{set}\big(\mathsf{in}^T\big) \to \mathsf{List}(N^T)$ given by $\mathsf{ch}^T(2) = [\,], \mathsf{ch}^T(3) = [1,2]$.

Note that the graphical representation of T is read bottom to top, with squares representing input nodes. Output nodes, on the other hand, are depicted by all those nodes that have outgoing wires towards the top of the diagram, not connected to any other node. In particular, the list of output nodes may contain duplicates, whereas the list of input nodes may not (the third of the axioms in Theorem III.5).

A. The sequential composition of term graphs

The reason we consider term graphs with multiple inputs and outputs, and attack tree components in particular, is that it allows us to understand the compositional structure of attack trees. There are two ways to compose term graphs: *sequential composition* and *parallel composition*.

We start with the former, an example for which was already given in (4). The sequential composition is given by gluing the output nodes of the first term graph to the input nodes of the second term graph. More formally:

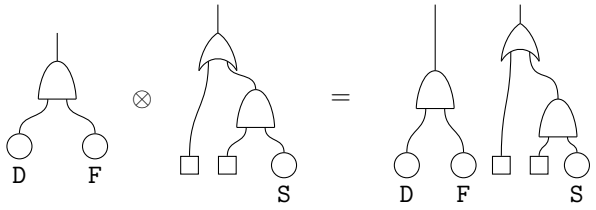
Definition III.9 (Sequential composition of term graphs)**.** Let $S\colon i \to j, T\colon j \to k$ be term graphs over a signature Σ. Their *(sequential) composition* is $T \circ S\colon i \to k$, where:

1) $N^{T\circ S} := \big(N^S \sqcup N^T\big) / \sim$. Here, $\sim$ is the equivalence relation that identifies the l-th output node of S with the l-th input node of T, for all $l \in \{1,\dots,j\}$.
2) $\mathsf{in}^{T\circ S} := \phi^S_*[\mathsf{in}^S]$ and $\mathsf{out}^{T\circ S} := \phi^T_*[\mathsf{out}^T]$, where $\phi^S\colon N^S \to N^{T\circ S}$, $\phi^T : N^T \to N^{T\circ S}$ are the inclusion maps followed by the canonical projection onto the quotient, and $f_*\colon \mathsf{List}(X) \to \mathsf{List}(Y)$ denotes the extension of a map $f : X \to Y$ between sets X, Y to lists.
3) $\mathsf{label}^{T\circ S}\colon N^{T\circ S} \setminus \mathsf{set}\big(\mathsf{in}^{T\circ S}\big) \to \Sigma$ is given by
$$\mathsf{label}^{T\circ S}([n]) := \begin{cases} \mathsf{label}^T(n) \text{ if } n \in N^T \setminus \mathsf{set}\big(\mathsf{in}^T\big) \\ \mathsf{label}^S(n) \text{ if } n \in N^S \setminus \mathsf{set}\big(\mathsf{in}^S\big), \end{cases}$$
for all $[n] \in N^{T\circ S}$
4) Similarly, $\mathsf{ch}^{T\circ S}\colon N^{T\circ S} \to \mathsf{List}(N^{T\circ S})$ is given by
$$\mathsf{ch}^{T\circ S}([n]) := \begin{cases} \mathsf{ch}^T(n) \text{ if } n \in N^T \setminus \mathsf{set}\big(\mathsf{in}^T\big) \\ \mathsf{ch}^S(n) \text{ if } n \in N^S \setminus \mathsf{set}\big(\mathsf{in}^S\big), \end{cases}$$
for all $[n] \in N^{T\circ S}$

Note that both $\mathsf{label}^{T\circ S}$ and $\mathsf{ch}^{T\circ S}$ are well-defined this way, by the definitions of $N^{T\circ S}$ and $\mathsf{in}^{T\circ S}$.

B. The parallel composition of term graphs

The parallel composition is given by the disjoint union, concatenating the respective lists of input and output nodes:



(5)

The full definition is as follows.

Definition III.10 (Parallel composition of term graphs)**.** Let $S\colon i \to j, T\colon k \to l$ be term graphs over a signature Σ. Their *parallel composition* is $T \otimes S\colon i + k \to j + l$, where:

1) $N^{T\otimes S} := N^S \sqcup N^T$.
2) $\mathsf{in}^{T\otimes S} := \mathsf{in}^S \cdot \mathsf{in}^T$ and $\mathsf{out}^{T\otimes S} := \mathsf{out}^S \cdot \mathsf{out}^T$.
3) $\mathsf{label}^{T\otimes S}\colon N^{T\otimes S} \setminus \mathsf{set}\big(\mathsf{in}^{T\otimes S}\big) \to \Sigma$ is given by
$$\mathsf{label}^{T\otimes S}(n) := \begin{cases} \mathsf{label}^T(n) \text{ if } n \in N^T \setminus \mathsf{set}\big(\mathsf{in}^T\big) \\ \mathsf{label}^S(n) \text{ if } n \in N^S \setminus \mathsf{set}\big(\mathsf{in}^S\big), \end{cases}$$
for all $n \in N^{T\otimes S}$.
4) Similarly, $\mathsf{ch}^{T\otimes S}\colon N^{T\otimes S} \to \mathsf{List}(N^{T\otimes S})$ is given by
$$\mathsf{ch}^{T\otimes S}(n) := \begin{cases} \mathsf{ch}^T(n) \text{ if } n \in N^T \setminus \mathsf{set}\big(\mathsf{in}^T\big) \\ \mathsf{ch}^S(n) \text{ if } n \in N^S \setminus \mathsf{set}\big(\mathsf{in}^S\big), \end{cases}$$
for all $n \in N^{T\otimes S}$.

C. Atomic term graphs

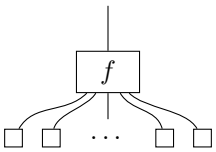
We will show that every term graph can be constructed, via parallel and sequential composition, from so-called *atomic* term graphs. These consist of term graphs of function symbols, plus a few structural ones which we describe below.

1) Identity term graphs: The simplest term graphs are the *identity term graphs*, $\mathrm{id}_0 \colon 0 \to 0$ and $\mathrm{id}_1 \colon 1 \to 1$. Here, id_0 is the empty term graph, and id_1 has exactly one node, which is both input and output:

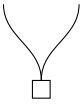
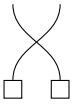
$$\mathrm{id}_1 \;:=$$ 

Note that id_0 is the identity for parallel composition, and the parallel composition of an appropriate amount of copies of id_1 is the identity for sequential composition.

2) Term graphs of function symbols: Each function symbol f in the signature Σ has an *associated term graph* $\langle f \rangle$ with $\mathsf{arity}(f) + 1$ nodes as follows:

$\langle f \rangle \;:=$ 

3) Copy, delete, swap: The term graphs $\mathsf{copy} \colon 1 \to 2$, $\mathsf{del} \colon 1 \to 0$, $\mathsf{swap} \colon 2 \to 2$ are the following term graphs:

$\mathsf{copy} \;:=$  $\mathsf{del} := \square$ $\mathsf{swap} :=$ 

In words, copy is a term graph with one input that is also two outputs, del has one input and no outputs, and swap has two inputs which are also outputs, but interchanged. Note that despite their names and appearance, these are not (yet) maps that duplicate inputs, delete inputs, and swap inputs; that will come later, when we interpret these terms graphs as operators in a channel category in Section IV.

Definition III.11. An *atomic term graph* over the signature Σ is a term graph E such that

$$E \in \{\mathrm{id}_0, \mathrm{id}_1, \mathsf{copy}, \mathsf{del}, \mathsf{swap}\} \cup \{\langle f \rangle \mid f \in \Sigma\}.$$

D. Decomposition into atomic term graphs

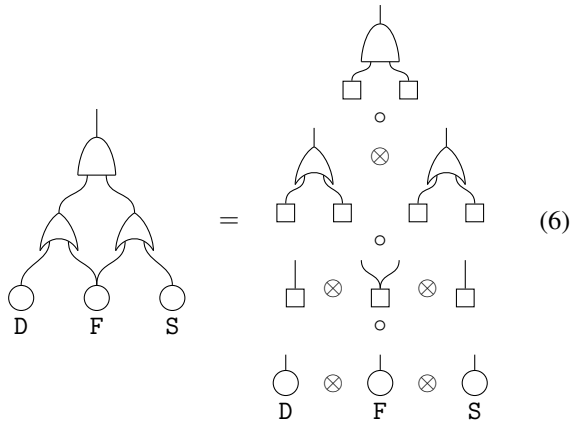
The following theorem is [CG99, Theorem 9].

Theorem III.12. *Every term graph is a sequential composite of parallel composites of atomic term graphs.*

In more detail, let T be a term graph over a signature Σ. Then there exist N, $k_1, \ldots, k_N \in \mathbb{Z}_{\geq 0}$, and atomic term graphs $A_{1,1}, \ldots, A_{k_1,1}, \ldots A_{1,N}, \ldots A_{k_N,N}$ over Σ such that

$$T = \bigcirc_{i=1}^{N} \bigotimes_{j=1}^{k_i} A_{i,j}.$$

Example III.13. Consider the attack tree T from the right hand side of Equation (4). It can be decomposed into atomic $\mathsf{AT}(B)$-term graphs as follows:

$T \;=\;$ 

(6)

We can also write this algebraically as,

$$\begin{aligned} T = {} & \mathtt{AND} \circ (\mathtt{OR} \otimes \mathtt{OR}) \\ & \circ (\mathrm{id}_1 \otimes \mathsf{copy} \otimes \mathrm{id}_1) \\ & \circ (\mathtt{D} \otimes \mathtt{F} \otimes \mathtt{S}), \end{aligned} \tag{7}$$

and, in fact, by Theorem III.12, every attack tree can be written in a similar form. Note that this decomposition is not unique, as we could equally write T as,

$$T = \mathtt{AND} \circ (\mathtt{OR} \otimes \mathtt{OR}) \circ (\mathtt{D} \otimes \mathsf{copy} \otimes \mathtt{S}) \circ \mathtt{F}, \tag{8}$$

or even arbitrarily reorder parallel composites by introducing swap term graphs.

IV. Semantics: Channels

So far, we have only discussed syntax. Now, we investigate semantics: what does a term graph *denote*?

To answer this question, we must first assign meaning to each function symbol in the given signature. This is formalised by the notion of an *interpretation*. From such an interpretation, we directly obtain semantics for the atomic term graphs associated to each function symbol. In order to lift these semantics to *general*, composite term graphs, we need to know the semantics of all further atomic term graphs, and, most importantly, how to compose them. This information is captured by the notion of a *channel category*: a structure consisting of sets of *channels*, together with rules for composing them sequentially and in parallel.

Hence, the answer to the question of the semantics of term graphs is that, in general, *term graphs denote channels*. Examples of channels include ordinary functions, stochastic matrices (modelling conditional probability distributions, or, "functions with random outputs"), and term graphs themselves. What a "channel" is, in general, we describe *axiomatically*, via the notion of a channel category.

Remark IV.1. *For readers familiar with the literature on* gs-monoidal categories *(see [Gad96], [CG99], [FL23]), we note that channel categories can be defined concisely as strict gs-monoidal categories generated by a single object.*

The following definition does not require any prior knowledge of category theory.

Definition IV.2 (Channel category)**.** A *channel category* C consists of the following, for all $i, j, k, l \in \mathbb{Z}_{\geq 0}$:

1) A set $C(i, j)$, whose elements we call *channels with i inputs and j outputs*.
2) A function $\circ\colon C(i, j) \times C(j, k) \to C(i, k)$ called *(sequential) composition* of channels.
3) A function $\otimes\colon C(i, j) \times C(k, l) \to C(i + k, j + l)$ called *parallel composition* of channels.
4) A number of special channels:
   a) $\mathsf{id}_i \in C(i, i)$, the *identity channels*,
   b) $\mathsf{swap}_{i,j} \in C(i + j, i + j)$, the *swap gates*.
   c) $\mathsf{copy} \in C(1, 2)$, the *copy gate*,
   d) $\mathsf{del} \in C(1, 0)$, the *delete* or *discard gate*,

These data are required to satisfy the following axioms, for all $i, j, k, l, m, n \in \mathbb{Z}_{\geq 0}$:

1) (Associativity and unitality of sequential composition) For all $f \in C(i, j)$, $g \in C(j, k)$, and $h \in C(k, l)$,
$$(h \circ g) \circ f = h \circ (g \circ f), \quad f \circ \mathsf{id}_i = f = \mathsf{id}_j \circ f.$$
2) (Associativity and unitality of parallel composition) For all $f \in C(i_1, j_1)$, $g \in C(i_2, j_2)$, and $h \in C(i_3, j_3)$,
$$(h \otimes g) \otimes f = h \otimes (g \otimes f), \quad f \otimes \mathsf{id}_0 = f = \mathsf{id}_0 \otimes f.$$
3) (Symmetry and functoriality of parallel composition) For all $f \in C(i, j)$ and $g \in C(k, l)$,
$$\mathsf{swap}_{l,j} \circ (g \otimes f) = f \otimes g = (g \otimes f) \circ \mathsf{swap}_{i,k},$$
$$\mathsf{swap}_{i,j} \circ \mathsf{swap}_{j,i} = \mathsf{id}_{i+j},$$
as well as, for all $f_1 \in C(i, j)$, $f_2 \in C(k, l)$, $g_1 \in C(j, m)$, and $g_2 \in C(l, n)$,
$$(g_1 \otimes g_2) \circ (f_1 \otimes f_2) = (g_1 \circ f_1) \otimes (g_2 \circ f_2).$$
4) (Commutative comonoid laws for copy and discard) The copy and discard gates satisfy the following equations:
$$(\mathsf{copy} \otimes \mathsf{id}_1) \circ \mathsf{copy} = (\mathsf{id}_1 \otimes \mathsf{copy}) \circ \mathsf{copy},$$
$$(\mathsf{del} \otimes \mathsf{id}_1) \circ \mathsf{copy} = \mathsf{id}_1 = (\mathsf{id}_1 \otimes \mathsf{del}) \circ \mathsf{copy},$$
$$\mathsf{copy} = \mathsf{swap}_{1,1} \circ \mathsf{copy}.$$

Example IV.3 ($\mathsf{TermGraphs}_\Sigma$)**.** Let Σ be a signature. The channel category $\mathsf{TermGraphs}_\Sigma$ of term graphs over Σ is given as follows. For each $i, j \in \mathbb{Z}_{\geq 0}$, the set of channels $\mathsf{TermGraphs}_\Sigma(i, j)$ is the set of term graphs with i inputs and j outputs. The sequential and parallel composition of term graphs are given as defined in Theorems III.9 and III.10. The identity channel id_i is the parallel composition of i copies of id_1. The copy and delete gates are the atomic term graphs copy and del. Finally, the swap gate $\mathsf{swap}_{i,j}$ is given by the term graph that swaps the first i inputs with the last j ones.

Example IV.4 ($\mathsf{Functions}_X$)**.** Let X be a set. Then the channel category $\mathsf{Functions}_X$ is given as follows. For each $i, j \in \mathbb{Z}_{\geq 0}$, the set of channels $\mathsf{Functions}_X(i, j)$ is the set of functions $X^i \to X^j$. The sequential and parallel composition are given by the ordinary composition of functions and the cartesian product of functions,
$$f \otimes g := f \times g, \quad (f \times g)(x, y) := (f(x), g(y))$$
respectively, and the identity channel id_i is the identity function on X^i, for each $i \in \mathbb{Z}_{\geq 0}$. The copy gate is given by,
$$\mathsf{copy} : X \to X^2, \ \mathsf{copy}(x) := (x, x),$$
and the delete gate is the unique map $X \to \mathbf{1}$ from X to the one-point set $\mathbf{1} = X^0$. Finally, the swap gate $\mathsf{swap}_{i,j}$ is given by the permutation $X^{i+j} \to X^{i+j}$ that swaps the first i coordinates with the last j ones. Note that $\mathsf{Functions}_X(0, 1)$ is the set of functions $\mathbf{1} \to X$, which may be identified with X itself.

A. Semantics

We now wish to define an *attack tree semantics* as a mapping from attack tree components to some channel category that preserves the compositional structure of attack trees, as motivated in Sections I and II. What it means to "preserve compositional structure" is formalised by the concept of a *functor of channel categories*.

Definition IV.5. Let C, D be channel categories. A *functor of channel categories* $F\colon C \to D$ is a family of functions $F\colon C(i, j) \to D(i, j)$ for each pair $i, j \in \mathbb{Z}_{\geq 0}$, such that the following equations hold for all channels on which they are defined:
$$F(g \circ f) = F(g) \circ F(f), \qquad F(\mathsf{id}_i) = \mathsf{id}_i,$$
$$F(g \otimes f) = F(g) \otimes F(f), \qquad F(\mathsf{swap}_{i,j}) = \mathsf{swap}_{i,j},$$
$$F(\mathsf{copy}) = \mathsf{copy}, \qquad F(\mathsf{del}) = \mathsf{del},$$

With this definition established, we can now present the definition that constitutes our solution to Theorem I.1.

Definition IV.6 (Attack tree semantics and metrics)**.** Let B be a set. A *semantics for* $\mathsf{AT}(B)$, or a *metric for* $\mathsf{AT}(B)$, is a pair (C, F) of a channel category C and a functor of channel categories $F\colon \mathsf{TermGraphs}_{\mathsf{AT}(B)} \to C$.

The advantage of this definition is that it is general enough to capture various metrics and semantics from the literature (see Section V). At the same time, the preservation of $\circ$ and $\otimes$ means that the decomposition of Theorem III.12 is preserved. When the target channel category is "sufficiently quantitative", such as the examples $\mathsf{Functions}$ and $\mathsf{Matrices}$ of Section II, this can be leveraged to obtain metric computation algorithms. We discuss this in detail in Theorem IV.12.

In our framework, *there is no difference between AT metrics and semantics*. Semantics are thought of as being more qualitative while metrics are more quantitative, but this distinction is not a mathematical one.

Our category-theoretical formulation can easily be applied to extensions of the AT framework, such as *dynamic ATs* [JKM$^+$15], attack-defence trees [KMRS10] and attack-fault trees [KS17], and fault tree extensions such as dynamic fault

trees [AKGP20], as these are all directed acyclic graphs with a finite number of gate types. Interpreting existing metrics of such frameworks as channel categories, as we do for standard ATs in Section V, is beyond the scope of this paper.

B. Interpretations

The downside of Definition IV.6 is that it does not explain how a functor of channel categories can be constructed. In this subsection, we show that such a functor is uniquely characterised by the image of the elements of its signature. The map from a signature to a channel category is called an *interpretation*.

Definition IV.7 (Interpretation of a signature)**.** Let Σ be a signature. An *interpretation* of Σ in a channel category C is a map

$$\mathcal{I}\colon \Sigma \to \bigsqcup_{i\in\mathbb{Z}_{\geq 0}} C(i,1)$$

such that for all $f \in \Sigma$, $\mathcal{I}(f) \in C(\mathsf{arity}(f), 1)$. Here, $\bigsqcup$ denotes the disjoint union of sets.

Example IV.8. Consider the signature $\mathsf{AT}(B)$ of attack trees over a set of basic attack step labels B from Theorem III.3. Let $t\colon B \to \{0,1\}$ be an assignment of truth values to each basic attack step label, which we interpret as indicating whether or not a basic attack step labelled with $b \in B$ is successful. For each such t, we may interpret the basic attack step labels as their corresponding truth values, and the logical connectives as their corresponding Boolean functions. This yields the following interpretation of $\mathsf{AT}(B)$ in the channel category $\mathsf{Functions}_{\{0,1\}}$:

$$\begin{aligned}
\mathcal{I}_t\colon \mathsf{AT}(B) &\to \mathsf{Functions}_{\{0,1\}} \\
\mathcal{I}_t(b)(\,) &:= t(b) && (b \in B) \\
\mathcal{I}_t(\mathtt{AND}_i)(x_1,\dots,x_i) &:= \bigwedge_{k=1}^{i} x_k && (i \in \mathbb{Z}_{\geq 0}) \\
\mathcal{I}_t(\mathtt{OR}_i)(x_1,\dots,x_i) &:= \bigvee_{k=1}^{i} x_k && (i \in \mathbb{Z}_{\geq 0})
\end{aligned}$$

These equations show that our notion of an interpretation is well-aligned with the standard notion from logic.

The following theorem says that there is a *unique* way to extend a given interpretation of any signature to a functor of channel categories; see [CG99, Theorem 23] for a proof in the setting of gs-monoidal categories.

Because term graphs can be decomposed into atomic graphs and channel category functors preserve composition, functors are uniquely defined by the interpretation of the signature:

Theorem IV.9. [CG99, Theorem 23] *Let Σ be a signature and let C be a channel category, and let $\mathcal{I}$ be an interpretation of Σ in C. Then there exists a unique functor of channel categories,*

$$[\![\,\cdot\,]\!]_{\mathcal{I}}\colon \mathsf{TermGraphs}_{\Sigma} \to C,$$

such that $[\![f]\!]_{\mathcal{I}} = \mathcal{I}(f)$ for all $f \in \Sigma$. Conversely, for every functor of channel categories $F\colon \mathsf{TermGraphs}_{\Sigma} \to C$ there exists an interpretation $\mathcal{I}$ such that $F = [\![\,\cdot\,]\!]_{\mathcal{I}}$.

This allows us to define the semantics of a term graph *under* an interpretation.

Definition IV.10 (Semantics under interpretation)**.** Let T be a term graph over a signature Σ, and let $\mathcal{I}$ be an interpretation of Σ in a channel category C. The *semantics* of T under $\mathcal{I}$ is defined to be $[\![T]\!]_{\mathcal{I}}$, i.e. the value at T of the unique functor of channel categories extending $\mathcal{I}$ to all term graphs.

Example IV.11. Let $B := \{\mathtt{D}, \mathtt{F}, \mathtt{S}\}$ be the set of basic attack step labels from Theorem III.4, let $t\colon B \to \{0,1\}$ be the assignment that maps each basic attack step label to a truth value,

$$\mathtt{D} \mapsto 1,\ \mathtt{F} \mapsto 0,\ \mathtt{S} \mapsto 1,$$

and let $\mathcal{I}_t$ be the Boolean function interpretation from Theorem IV.8. What are the semantics $[\![T]\!]_{\mathcal{I}_t}$ of the attack tree T from Theorem III.13 under $\mathcal{I}_t$?

Since part of the defining property of the semantics under an interpretation is that it preserves compositional structure (including copy gates), we first use the decomposition (8) from Theorem III.13, and then plug in the definitions:

$$\begin{aligned}
[\![T]\!]_{\mathcal{I}_t} &= [\![\mathtt{AND} \circ (\mathtt{OR} \otimes \mathtt{OR}) \circ (\mathtt{D} \otimes \mathsf{copy} \otimes \mathtt{S}) \circ \mathtt{F}]\!]_{\mathcal{I}_t} \\
&= [\![\mathtt{AND}]\!]_{\mathcal{I}_t} \circ ([\![\mathtt{OR}]\!]_{\mathcal{I}_t} \otimes [\![\mathtt{OR}]\!]_{\mathcal{I}_t}) \\
&\quad \circ ([\![\mathtt{D}]\!]_{\mathcal{I}_t} \otimes \mathsf{copy} \otimes [\![\mathtt{S}]\!]_{\mathcal{I}_t}) \circ [\![\mathtt{F}]\!]_{\mathcal{I}_t} \\
&= \mathcal{I}_t(\mathtt{AND}) \circ (\mathcal{I}_t(\mathtt{OR}) \otimes \mathcal{I}_t(\mathtt{OR})) \\
&\quad \circ (\mathcal{I}_t(\mathtt{D}) \otimes \mathsf{copy} \otimes \mathcal{I}_t(\mathtt{S})) \circ \mathcal{I}_t(\mathtt{F}) \\
&= \mathcal{I}_t(\mathtt{AND})(\mathcal{I}_t(\mathtt{OR})(\mathcal{I}_t(\mathtt{D}),\ \mathcal{I}_t(\mathtt{F})), \\
&\qquad\qquad \mathcal{I}_t(\mathtt{OR})(\mathcal{I}_t(\mathtt{F}),\ \mathcal{I}_t(\mathtt{S}))) \\
&= (1 \vee 0) \wedge (0 \vee 1) = 1.
\end{aligned}$$

Hence, under the given interpretation, T evaluates to "true", as we would expect. This example shows once more that our notion of interpretation generalises the standard notion from propositional logic.

The approach in this example works more generally: we can decompose an AT according to Theorem III.12, interpret the resulting components using $\mathcal{I}$, and then put them back together using $\circ$ and $\otimes$ on the channel category side.

Theorem IV.12. *Let T be an AT, and let $T = \bigcirc_{i=1}^{N} \bigotimes_{j=1}^{k_i} A_{i,j}$ be its decomposition into atomic AT components. Let $\mathcal{I}$ be an interpretation of $\mathsf{AT}(B)$ in a channel category C. Then*

$$[\![T]\!]_{\mathcal{I}} = \bigcirc_{i=1}^{N} \bigotimes_{j=1}^{k_i} [\![A_{i,j}]\!]_{\mathcal{I}}.$$

Effectively, this is a metric computation algorithm, provided that $\circ, \otimes$ are computationally tractable; our decomposition result follows from [CG99], which has a constructive proof. This approach is akin to Bayesian network analysis via string diagrams [Jac18], which maps the graphical structure of a

Bayesian network to the corresponding matrix operations. In Section V, we discuss how this result can be used to compute specific metrics.

Note that in an interpretation $\mathcal{I}$, there needs to be no connection between $\mathcal{I}(\mathtt{AND}_i)$ and $\mathcal{I}(\mathtt{AND}_j)$, and the operators $\mathcal{I}(\mathtt{AND}_i)$ need not be 'commutative' in the sense that $\mathcal{I}(\mathtt{AND}_{i+j}) \circ \mathsf{swap}_{i,j} = \mathcal{I}(\mathtt{AND}_{i+j})$. One could demand such restrictions, but we leave these out for now to be able to apply our framework to extensions of ATs, which typically have non-commutative structure.

V. Application to existing metric frameworks

Up to this point, we have provided a compositional theory of the syntax and semantics of attack trees on a general level. What we still need to demonstrate is that all common attack tree metrics can indeed be viewed as instances of these general compositional semantics. To this end, we show that all these attack tree metrics correspond to the semantics of attack trees under a certain interpretation in a channel category. More concretely, we give channel category interpretations for the following metrics and semantics:

- The bottom-up semiring metrics of [MO05];
- The propositional semiring metrics of [LZBS22];
- Fault tree unreliability [RS15];
- The multiset semantics of [MO05];
- The propositional semantics of [LZBS22].

Together, these cover most metrics used in the literature (with only few exceptions, as discussed in Section V-F); note that the set-semantics metrics of [KPCS14] coincide with the propositional metrics when the underlying semiring is absorbing (see below), which is almost always the case [LZBS22].

Within both the bottom-up and propositional metric frameworks, there exist many relevant metrics, such as attack time, cost, skill, probability, etc. Both frameworks encode these by a set R of possible metric values, and two binary operators $+$ and $\cdot$, corresponding to OR and AND gates, respectively. Each basic attack step label b is assigned a metric value $\alpha(b) \in R$, and the map $\alpha\colon B \to R$ is called an *attribution*. How these together define the AT's metric value differs per framework, as discussed in Section II. Nevertheless, both methodologies require the tuple $(R, +, \cdot)$ to satisfy the algebraic property that it forms a semiring:

Definition V.1 (Semiring)**.** A *semiring* is a set R equipped with two binary operations $+$ and $\cdot$, and two elements 0 and 1, such that $+$ and $\cdot$ are binary associative commutative operations on R with unit 0 and 1, respectively, and such that $\cdot$ distributes over $+$, i.e.

$$r \cdot (s + t) = r \cdot s + r \cdot t$$

for all $r, s, t \in R$. If furthermore $r + (r \cdot s) = r$ for all r, s, we call R *absorbing*.

An overview of metrics from the literature is given in Table I. All are absorbing except *max challenge* and fault tree *unreliability*.

TABLE I
Overview of semirings $(R, +, \cdot, 0, 1)$.

Type of metric	R	$+$	$\cdot$	0	1
Min. cost	$[0, \infty]$	min	$+$	∞	0
Min. time (parallel)	$[0, \infty]$	min	max	∞	0
Min. time (sequential)	$[0, \infty]$	min	$+$	∞	0
Max. challenge	$[0, \infty]$	max	max	∞	∞
Max. probability	$[0, 1]$	max	$\cdot$	0	1
Unreliability (fault trees)	$[0, \infty)$	$+$	$\cdot$	0	1

A. Bottom-up semiring metrics

We begin with the case of bottom-up metrics.

Definition V.2. Let B be a set, let R be a semiring and let $\alpha : B \to R$ be an attribution. The *bottom-up interpretation* $\mathcal{I}^\alpha_{\mathsf{bu}}$ is defined as follows:

$$\begin{aligned} \mathcal{I}^\alpha_{\mathsf{bu}} &\colon \mathsf{AT}(B) \to \mathsf{Functions}_R, \\ \mathcal{I}^\alpha_{\mathsf{bu}}(\mathtt{AND}_i)(x_1, \ldots, x_i) &:= \prod_{k=1}^{i} x_k, \\ \mathcal{I}^\alpha_{\mathsf{bu}}(\mathtt{OR}_i)(x_1, \ldots, x_i) &:= \sum_{k=1}^{i} x_k, \\ \mathcal{I}^\alpha_{\mathsf{bu}}(b)(\,) &:= \alpha(b). \qquad (b \in B) \end{aligned}$$

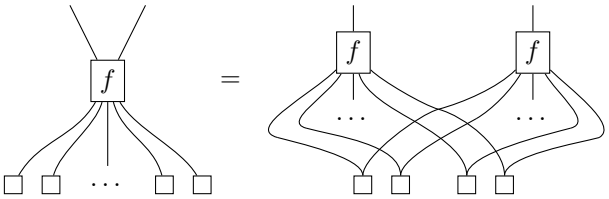
The reason we call this the *bottom-up* interpretation is that the semantics under this interpretation recover exactly the *bottom-up* semiring metrics of [MO05]. Their name comes from the following application of Theorem IV.12, which essentially states that the semantics of an an attack tree T under $\mathcal{I}^\alpha_{\mathsf{bu}}$ can be computed recursively (i.e. "bottom-up") on the structure of T.

Theorem V.3. *Let B be a set, let R be a semiring and let $\alpha : B \to R$ be an attribution. Moreover, let T be an attack tree over B whose unique output R_T has n children, and let T'_i be the attack tree rooted at the i-th child of R_T, for all $i \in \{1, \ldots, n\}$. Then:*

$$[\![T]\!]_{\mathcal{I}^\alpha_{\mathsf{bu}}} = \begin{cases} \alpha(b) & \text{if } \mathsf{label}(R_T) = b, b \in B, \\ \sum_{i=1}^{n} [\![T'_i]\!]_{\mathcal{I}^\alpha_{\mathsf{bu}}} & \text{if } \mathsf{label}(R_T) = \mathtt{OR}_n, \\ \prod_{j=1}^{n} [\![T'_i]\!]_{\mathcal{I}^\alpha_{\mathsf{bu}}} & \text{if } \mathsf{label}(R_T) = \mathtt{AND}_n. \end{cases}$$

Proof. When T is tree-structured (i.e. every node has a unique parent node), the claim follows directly by induction on the tree structure of T, applying Theorem IV.12 successively. For the general case, we may rewrite T into a tree-structured attack tree in a way that preserves the semantics under $\mathcal{I}^\alpha_{\mathsf{bu}}$, since the interpretation of all function symbols is

deterministic, i.e. all function symbols f satisfy



in the semantics under $\mathcal{I}^{\alpha}_{\mathsf{bu}}$. □

Example V.4. In the *min cost* metric, semiring addition is $\min$, and semiring multiplication is $+$. Hence, as in Section II, an AND-gate adds the costs of its children, since all need to be activated; and an OR-gate selects the minimum cost of its children, as only one needs to be activated.

B. The stochastic matrix interpretation of an attack tree

In order to treat the case of propositional semiring metrics and fault tree unreliability in a uniform way, we first introduce the *stochastic matrix interpretation*. This makes use of the notion of a *Boolean-indexed stochastic matrix* over a semiring. Intuitively, a Boolean-indexed stochastic matrix can be understood as a (generalised) conditional probability table, describing the probabilities (or cost, time, etc.) of transitioning from one vector of Booleans to another, possibly of a different size.

Definition V.5. Let R be a semiring. A *Boolean-indexed matrix* over R is a $2^j \times 2^i$-matrix over R for some $i, j \in \mathbb{Z}_{\geq 0}$. Here, we write $2 = \{0, 1\}$ for the set of Booleans, and thus view such matrices as indexed over vectors of Booleans. In particular, vectors $v \in R^2$ will be indexed as $(v_0, v_1)^\top$, and hence, **in the following, v_1 will always denote the *second* component of v.**

A $2^j \times 2^i$-matrix A over R is *stochastic* if for all $x \in 2^i$,

$$\sum_{y \in 2^j} A_{yx} = 1. \tag{9}$$

When $R = ([0, \infty), +, \cdot, 0, 1)$ is the semiring of nonnegative real numbers with their ordinary addition and multiplication, and A is square, this recovers the usual notion of a column-stochastic matrix.

The entry $A_{yx} \in R$ can be thought of as the probability of – or cost, or time, etc., required to – transition from the "state" x to the "state" y, with both of these "states" being vectors of Booleans of possibly different sizes.

Whenever we write a Boolean-indexed stochastic matrix P in tabular form, as in

$$P = \begin{pmatrix} 0.7 & 0.1 & 0.2 & 0.6 \\ 0.3 & 0.9 & 0.8 & 0.4 \end{pmatrix} \in [0, \infty)^{2^1 \times 2^2},$$

we order the indices of P according to their binary-integer value. For example, the entry $P_{(1),\,(1\,0)} = 0.8$ is the entry in the second row and third column, and represents the probability that the input $(0\ 1)$ transitions into the output 1.

The stochastic matrix interpretation now takes values in the following channel category.

Definition V.6 ($\mathsf{BoolStoch}_R$)**.** Let R be a semiring. The channel category $\mathsf{BoolStoch}_R$ of Boolean-indexed stochastic matrices over R is given as follows. For each $i, j \in \mathbb{Z}_{\geq 0}$, the set of channels $\mathsf{BoolStoch}_R(i, j)$ is the set of stochastic $2^j \times 2^i$-matrices over R. The sequential composition of stochastic matrices is given by matrix multiplication, while the parallel composition is given by the Kronecker product of matrices. The identity channel id_i is the $2^i \times 2^i$-identity matrix, and the copy and delete gates are given by the following matrices:

$$\mathsf{copy} := \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}, \quad \mathsf{del} := \begin{pmatrix} 1 & 1 \end{pmatrix}.$$

Finally, the swap gate $\mathsf{swap}_{i,j}$ is given by the permutation matrix that swaps the first i coordinates with the last j ones.

Remark V.7. *The requirement* (9) *of working with stochastic matrices ensures that* $\mathsf{BoolStoch}_R$*, like* $\mathsf{Functions}_R$*, is a* Markov category *[Fri20]. This means that the only morphism with no output is the* del *channel ("discard").*

Definition V.8 (Stochastic matrix interpretation)**.** Let B be a set, let R be a semiring, and let $\alpha = (\alpha_1, \alpha_0)$ be a pair of functions $B \to R$ such that $\alpha_0(b) + \alpha_1(b) = 1$ for all $b \in B$. The *stochastic matrix interpretation* $\mathcal{I}^{\alpha}_{\mathsf{stoch}}$ is defined as follows:

$$\mathcal{I}^{\alpha}_{\mathsf{stoch}} \colon \mathsf{AT}(B) \to \mathsf{BoolStoch}_R,$$

$$\mathcal{I}^{\alpha}_{\mathsf{stoch}}(\mathtt{AND}_i) := \begin{pmatrix} 1 & 1 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \in R^{2 \times 2^i},$$

$$\mathcal{I}^{\alpha}_{\mathsf{stoch}}(\mathtt{OR}_i) := \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 1 & 1 \end{pmatrix} \in R^{2 \times 2^i},$$

$$\mathcal{I}^{\alpha}_{\mathsf{stoch}}(b) := \begin{pmatrix} \alpha_0(b) \\ \alpha_1(b) \end{pmatrix}.$$

To relate this interpretation to propositional semiring metrics and fault tree unreliability, we first need to introduce some terms used to define the latter.

First, a *basic attack step* in an attack tree component T over a set B of basic attack step labels is simply any node labelled with an element of B.

Definition V.9 (Basic attack step)**.** Let T be an attack tree component over a set B. A basic attack step is a node $n \in N^T$ such that $\mathsf{label}^T(n) \in B$. We write $\mathsf{BAS}(T)$ for the set of basic attack steps of T.

Next, an *attack* indicates for each basic attack step whether it is performed or not.

Definition V.10 (Attack)**.** Let T be an attack tree component. An *attack* is a function $a \colon \mathsf{BAS}(T) \to \{0, 1\}$ from the set of basic attack steps to the set of Booleans. We write $\mathsf{Attacks}(T)$ for the set of all attacks.

Now, the *structure function* of an attack tree determines for each attack whether it is successful or not. More generally, the

structure function of an attack tree component determines the outcome of each output, for each attack and each additional input.

Definition V.11 (Structure function)**.** Let $T : i \to j$ be an attack tree component. The *structure function* of T is the function

$$S_T \colon \mathsf{Attacks}(T) \times \{0,1\}^i \to \{0,1\}^j,$$
$$S_T(a,x) := \left(S_T(a,x;\, \mathsf{out}_k^T)\right)_{k=1}^{j},$$

where for all $n \in N^T$, $S_T(a,x;\, n)$ is defined recursively as

$$S_T(a,x;\, n) := \begin{cases} x_k & \text{if } n = \mathsf{in}_k^T, \\ a(n) & \text{if } n \in \mathsf{BAS}(T), \\ \bigwedge_{m \in \mathsf{ch}^T(n)} S_T(a,x;\, m) & \\ & \text{if } \mathsf{label}^T(n) = \mathtt{AND}, \\ \bigvee_{m \in \mathsf{ch}^T(n)} S_T(a,x;\, m) & \\ & \text{if } \mathsf{label}^T(n) = \mathtt{OR}. \end{cases}$$

When $T : 0 \to 1$ is an attack tree, it has no input nodes, and the structure function is a simply function

$$S_T : \mathsf{Attacks}(T) \to \{0,1\}.$$

Example V.12. Consider the AT T of Example III.13. Let $n_{\mathtt{D}}$, $n_{\mathtt{F}}$, $n_{\mathtt{S}}$ be the nodes with labels D, F, S, respectively. An attack on T is a map $\{n_{\mathtt{D}}, n_{\mathtt{F}}, n_{\mathtt{S}}\} \to \{0,1\}$, which we may view as a three-dimensional binary vector $\vec{x} = (x_{\mathtt{D}}, x_{\mathtt{F}}, x_{\mathtt{S}})$. In Example IV.11, we saw that $S_T(101) = 1$. Using the same decomposition, but keeping the basic attack steps as variables, we find

$$S_T(\vec{x}) = (x_{\mathtt{D}} \vee x_{\mathtt{F}}) \wedge (x_{\mathtt{D}} \vee x_{\mathtt{S}}).$$

To keep the subsequent formulas concise, we will need one additional notational convenience.

Notation V.13. Let T be an attack tree component over a set B and let $\alpha : B \to R$ be an attribution (i.e. a function from B to some semiring R). We define

$$\overline{\alpha} : \mathsf{BAS}(T) \to R, \quad \overline{\alpha}_i(v) := \alpha_i(\mathsf{label}^T(v)).$$

We are now ready to state an explicit formula for the stochastic matrix interpretation in terms of the notions introduced above. We will use this result to show that both the propositional semiring metrics of [LZBS22] and the fault tree unreliability [RS15] can be viewed as semantics in a channel category. A proof is given in Section A.

Theorem V.14. *Let $T : i \to j$ be an attack tree component over a set B. Moreover, let R be a semiring, and let $\alpha = (\alpha_0, \alpha_1)$ be a pair of two functions $B \to R$ such that $\alpha_0(b) + \alpha_1(b) = 1$ for all $b \in B$. Then*

$$[\![T]\!]_{\mathcal{I}^\alpha_{\mathsf{stoch}}} = (M^T_{yx})_{y \in 2^j,\, x \in 2^i},$$

where

$$M^T_{yx} = \sum_{\substack{a \in \mathsf{Attacks}(T) \\ S_T(a,x) = y}} \left(\prod_{a(v)=1} \overline{\alpha}_1(v)\right) \cdot \left(\prod_{a(v)=0} \overline{\alpha}_0(v)\right).$$

When T is an attack tree, the statement of Theorem V.14 can be simplified, using the following notation for the set of all successful attacks.

Definition V.15 (Successful attacks)**.** Let T be an attack tree. The set of all successful attacks is denoted by

$$\mathsf{Suc}_T := S_T^{-1}(\{1\}) = \{a \in \mathsf{Attacks}(T) \mid S_T(a) = 1\}.$$

The following is now a direct consequence of Theorem V.14.

Corollary V.16. *Let T be an attack tree over a set B. As before, let R be a semiring, and let $\alpha = (\alpha_1, \alpha_0)$ be a pair of two functions $B \to R$ such that $\alpha_0(b) + \alpha_1(b) = 1$ for all $b \in B$. Then*

$$\left([\![T]\!]_{\mathcal{I}^\alpha_{\mathsf{stoch}}}\right)_1 = \sum_{a \in \mathsf{Suc}_T} \left(\prod_{a(v)=1} \overline{\alpha}_1(v)\right) \cdot \left(\prod_{a(v)=0} \overline{\alpha}_0(v)\right)$$

where the left-hand side is the second *component of the stochastic matrix semantics of T (a vector in R^2).*

Example V.17. Consider again the minimal cost semiring $([0,\infty], \min, +, \infty, 0)$. The restriction on α is now that $\min(\alpha_0(b), \alpha_1(b)) = 0$ for all b. We take $\alpha_0(b) = 0$, and $\alpha_1(b)$ to be the actual cost of any basic attack step labelled b: then Theorem V.16 tells us that

$$\begin{aligned} \left([\![T]\!]_{\mathcal{I}^\alpha_{\mathsf{stoch}}}\right)_1 &= \min_{a \in \mathsf{Suc}_T} \left(\sum_{a(v)=1} \overline{\alpha}_1(v) + \sum_{a(v)=0} \overline{\alpha}_0(v)\right) \\ &= \min_{a \in \mathsf{Suc}_T} \sum_{a(v)=1} \overline{\alpha}_1(v). \end{aligned}$$

This is indeed *min cost* as defined in [LZBS22]. The next subsection applies this approach to general semiring metrics.

C. Propositional semantics and metrics

Using Theorem V.16, we now show that the propositional semiring metrics of [LZBS22] are given by the semantics under a certain interpretation, the *propositional interpretation*. Hence, in particular, these semantics extend to arbitrary attack tree components, preserving the compositional structure of attack trees. In this section, we assume all semirings are *absorbing*.

The propositional interpretation is now defined as follows:

Definition V.18. Let B be a set, let R be an absorbing semiring, and let $\alpha \colon B \to R$ be an attribution. The *(α-weighted) propositional interpretation* $\mathcal{I}^\alpha_{\mathsf{prop}}$ is

$$\mathcal{I}^\alpha_{\mathsf{prop}} := \mathcal{I}^{\alpha'}_{\mathsf{stoch}},$$

where $\alpha' = (\alpha'_0, \alpha'_1)$ is the pair of functions $B \to R$ given by:

$$\alpha'_0(b) := \alpha(b), \;\; \alpha'_1(b) := 1. \qquad (b \in B)$$

The definition of propositional semiring metrics depends on the notion of *minimal successful attacks* of an AT T.

Definition V.19. Let T be an attack tree over a set B. Identifying attacks $B \to \{0,1\}$ with subsets of B, the set of *minimal successful attacks* of T is defined as

$$\mathsf{MinSuc}_T := \{a \in \mathsf{Suc}_T \mid \nexists a' \in \mathsf{Suc}_T : a' \subsetneq a\}.$$

In other words, MinSuc_T is the set of all successful attacks that are minimal in the sense that no other successful attack is a subset of them.

As we will show later, the set of minimal successful attacks can also be obtained as the semantics of T under a certain interpretation.

The following corollary of Theorem V.16 provides an explicit expression for the semantics of an attack tree T under the propositional interpretation $\mathcal{I}^{\alpha}_{\mathsf{prop}}$, coinciding precisely with the definition of propositional semiring metrics in [LZBS22].

Corollary V.20. *Let B be a set, let R be an absorbing semiring, and let $\alpha : B \to R$ be an attribution. Then for any attack tree T, we have:*

$$\left(\llbracket T \rrbracket_{\mathcal{I}^{\alpha}_{\mathsf{prop}}}\right)_1 = \sum_{a \in \mathsf{MinSuc}_T} \prod_{a(v)=1} \overline{\alpha}(v).$$

Proof. By the definition of $\mathcal{I}^{\alpha}_{\mathsf{prop}}$ and Theorem V.16,

$$\left(\llbracket T \rrbracket_{\mathcal{I}^{\alpha}_{\mathsf{prop}}}\right)_1 = \sum_{a \in \mathsf{Suc}_T} \left(\prod_{a(v)=1} \overline{\alpha}_1(v) \right). \tag{10}$$

Since R is absorbing, adding the product of some superset $a' \supseteq a$ of $a \in \mathsf{Suc}_T$ does not change the sum, and hence, replacing Suc_T by MinSuc_T in Equation (10) we obtain the desired identity. □

Theorem IV.12 now gives an algorithm to compute propositional semiring metrics, through repeated multiplication and Kronecker multiplication of matrices. This is completely different from existing algorithms based on binary decision diagrams [LZBS22] and clone deletion [KPCS14]. Since the size of the involved matrices are exponential in the number of inputs/outputs, to implement this effectively the decomposition of Theorem III.12 should be chosen such that the number of inputs/outputs at each composition is minimal. Optimising this is beyond the scope of this paper.

D. Qualitative attack tree semantics

In addition to attack tree metrics, qualitative semantics have also been proposed. We will discuss two such semantics, the *propositional semantics* [LZBS22] given by the set of minimal attacks, and the *multiset semantics* [MO05]. These are simply referred to as "the semantics of an attack tree" in these works. In contrast, we do not make a choice as to which qualitative semantics is the default, and also consider attack tree *metrics* as a type of (quantitative) semantics.

1) Minimal successful attacks: The set of minimal successful attacks MinSuc_T of an attack tree T (see Theorem V.19) can be realised as the semantics of T under the interpretation $\mathcal{I}^{\alpha}_{\mathsf{prop}}$, for a specific semiring $\mathsf{AC}(T)$ which we now construct.

For any attack tree T over a set B, we consider elements $a \colon \mathsf{BAS}(T) \to \{0,1\}$ of $\mathsf{Attacks}(T)$ as subsets of $\mathsf{BAS}(T)$, by considering them as characteristic functions. We then let

$$\mathsf{AC}(T) := \{A \subseteq \mathsf{Attacks}(T) \mid \forall a, a' \in A : a \not\subseteq a' \wedge a' \not\subseteq a\}$$

be the set of antichains in $\mathsf{Attacks}(T)$. The set $\mathsf{AC}(T)$ becomes a semiring under the following operations:

$$A + A' := (A \cup A')^{\mathsf{AC}}, \quad A \cdot A' := (\{a \cup a' \mid a \in A, a' \in A'\})^{\mathsf{AC}},$$

where

$$S^{\mathsf{AC}} := \{a \in S \mid \forall a' \in S : a' \subsetneq a\}$$

associates to S its antichain of elements that are minimal with respect to $\subseteq$. The additive and multiplicative units of $\mathsf{AC}(T)$ are given by the empty set $\emptyset$ and the singleton set $\{\emptyset\}$, respectively. Moreover, $\mathsf{AC}(T)$ is absorbing. (In fact, for finite B, $\mathsf{AC}(T)$ is the *free distributive lattice* on $\mathsf{BAS}(T)$.)

The following theorem now shows that the set of minimal attacks for an attack tree T is equivalently given by the semantics of T under the propositional interpretation with respect to a certain attribution in $\mathsf{AC}(T)$. The proof is an application of Theorem V.20 and given in Section B.

Theorem V.21. *Let T be an attack tree over a set B, and assume that there is no distinction between basic attack steps and their labels, in the sense that $B = \mathsf{BAS}(T)$ and $\mathsf{label}^T(v) = v$ for all $v \in \mathsf{BAS}(T)$. Let*

$$\alpha \colon B \to \mathsf{AC}(T), \quad b \mapsto \{\{b\}\},$$

be the attribution that assigns to each basic attack step b the singleton antichain $\{\{b\}\}$. Then

$$\mathsf{MinSuc}_T = \left(\llbracket T \rrbracket_{\mathcal{I}^{\alpha}_{\mathsf{prop}}}\right)_1.$$

Note that the assumption that there is no distinction between basic attack steps and labels is not restrictive, as this is an inherent feature of how attack trees are formalised in context of the propositional semantics as treated in [LZBS22].

2) The multiset of successful attacks: A different approach to qualitative semantics of attack trees are the *multiset semantics*, which are simply called "the semantics of an attack tree" in [MO05]. By definition, these are the bottom-up semantics associated to the following semiring.

For any set B let $\mathsf{M}(2^B)$ be the set of multisets of subsets of B. The set $\mathsf{M}(2^B)$ becomes a semiring under the following operations:

$$A + A' := A \cup A', \quad A \cdot A' := \{a \cup a' \mid a \in A, a' \in A\}.$$

where $\cup$ denotes the union of multisets, adding up multiplicities, and the multiset comprehension notation on the right-hand side similarly keeps track of multiplicities. The additive and multiplicative units of $\mathsf{M}(2^B)$ are given by the empty set $\emptyset$ and the singleton set $\{\emptyset\}$, respectively.

Under the multiset semantics, an attack step with multiple parents in the attack tree is interpreted as being performed repeatedly, once for each parent. In particular, in contrast to the semantics in terms of minimal attacks, sharing of attack steps in the attack tree does not affect these semantics.

E. Fault tree unreliability

A further important consequence of Theorem V.16 is the case of fault tree unreliability, the probability that the system represented by the fault tree fails.

Definition V.22. Let $R := [0, \infty)$ be the semiring of non-negative reals, with its usual addition and multiplication, and let B be a set, whose elements we think of as *basic events*. Moreover, let $\alpha \colon B \to R$ be an attribution such that $\alpha(b) \in [0, 1]$ for all $b \in B$. (This is usually called a *probabilistic status vector* in the context of fault trees.) The *unreliability interpretation* is the following interpretation of the signature $\mathsf{FT}(B) := \mathsf{AT}(B)$ of fault trees over B:

$$\mathcal{I}_{\mathsf{unrel}} \colon \mathsf{FT}(B) \to \mathsf{BoolStoch}_R,$$
$$\mathcal{I}_{\mathsf{unrel}} := \mathcal{I}_{\mathsf{stoch}}^{\alpha'},$$

where $\alpha' = (\alpha'_0, \alpha'_1)$ is the pair of functions $B \to R$:

$$\alpha'_0(b) = 1 - \alpha(b), \quad \alpha'_1(b) = \alpha(b). \qquad (b \in B)$$

With this definition in place, we obtain:

Corollary V.23. *Let T be an attack tree (thought of as a fault tree) over some set B ("basic events"), and let $\alpha \colon B \to [0, 1]$ ("probabilistic status vector"). Moreover, let $(X_v)_{v \in \mathsf{BAS}(T)}$ be independent random variables, each Bernoulli-distributed with parameter $p_v = \overline{\alpha}(v)$. Then*

$$(\llbracket T \rrbracket_{\mathcal{I}_{\mathsf{unrel}}})_1 = \mathbb{P}\left[S_T\left((X_v)_{v \in \mathsf{BAS}(T)}\right) = 1\right],$$

where S_T is the structure function of T. In other words, the semantics of T under the unreliability interpretation correspond to precisely to the probability that the system T fails.

Proof. According to Theorem V.16, we have:

$$(\llbracket T \rrbracket_{\mathcal{I}_{\mathsf{unrel}}})_1 = \sum_{a \in \mathsf{Suc}_T} \left(\prod_{a(v)=1} p_v \right) \cdot \left(\prod_{a(v)=0} (1 - p_v) \right).$$

Since, by definition, $\mathsf{Suc}_T = S_T^{-1}(\{1\})$, the expression on the right hand side is precisely the probability we need. □

Applying Theorem IV.12 to fault tree unreliability yields exactly the matrix-based analysis of Bayesian network of [Jac18]; note that Bayesian networks generalise fault trees.

F. Non-examples

Up to this point, we have shown which attack tree metrics and qualitative semantics arise as special cases of Theorem IV.6. We now turn to the converse question: which metrics or qualitative semantics do *not* naturally fit into our framework? Since metrics that preserve the channel category structure of attack tree components must also preserve the coarser modular composition considered in [LZ24], the non-examples of *operad metrics* given therein also apply in our case. A concrete example is given by the *mean time to compromise* [MBFB06]; see [LZ24] for details.

VI. Conclusion

The core of our results can be summarised in one slogan: *attack tree metrics are functors of channel categories*. This characterisation allows one to define and analyse attack tree metrics in a modular manner, by how they behave on the basic building blocks of attack trees. Moreover, it fruitfully identifies attack trees as string diagrams, thereby connecting them to the numerous other areas, see [PZ25], in which string diagrams have been applied. An interesting question for future research is how to effectively exploit the compositionality of attack tree metrics for algorithmic purposes in practice. Theorem IV.12 is a first qualitative step in this direction. However, more research is needed to design generic algorithms at this level that account for efficiency, and to evaluate their complexity and practical performance against specialised methods.

References

[AKGP20] Koorosh Aslansefat, Sohag Kabir, Youcef Gheraibia, and Yiannis Papadopoulos. Dynamic fault tree analysis: state-of-the-art in modeling, analysis, and tools. *Reliability management and engineering*, pages 73–112, 2020.

[BCJ11] Jacob D Biamonte, Stephen R Clark, and Dieter Jaksch. Categorical tensor network states. *AIP Advances*, 1(4), 2011.

[BHK+14] Kristian Beckers, Maritta Heisel, Leanid Krautsevich, Fabio Martinelli, Rene Meis, and Artsiom Yautsiukhin. Determining the probability of smart grid attacks by combining attack tree and attack graph analysis. In *International Workshop on Smart Grid Security*, pages 30–47. Springer, 2014.

[BK17] Angèle Bossuat and Barbara Kordy. Evil twins: Handling repetitions in attack–defense trees: A survival guide. In *International Workshop on Graphical Models for Security*, pages 17–37. Springer, 2017.

[CG99] Andrea Corradini and Fabio Gadducci. An algebraic presentation of term graphs, via gs-monoidal categories. *Applied Categorical Structures*, 7:299–331, 1999.

[CJ19] Kenta Cho and Bart Jacobs. Disintegration and Bayesian inversion via string diagrams. *Mathematical Structures in Computer Science*, 29(7):938–971, 2019.

[CK18] Bob Coecke and Aleks Kissinger. Picturing quantum processes: A first course on quantum theory and diagrammatic reasoning. In *International conference on theory and application of diagrams*, pages 28–31. Springer, 2018.

[DWT17] Huiyu Dong, Hongwei Wang, and Tao Tang. An attack tree-based approach for vulnerability assessment of communication-based train control systems. In *2017 Chinese Automation Congress (CAC)*, pages 6407–6412. IEEE, 2017.

[FL23] Tobias Fritz and Wendong Liang. Free gs-monoidal categories and free Markov categories. *Applied Categorical Structures*, 31(2):21, 2023.

[Fon13] Brendan Fong. Causal theories: A categorical perspective on Bayesian networks. *arXiv preprint arXiv:1301.6201*, 2013.

[Fri20] Tobias Fritz. A synthetic approach to Markov kernels, conditional independence and theorems on sufficient statistics. *Advances in Mathematics*, 370, 2020.

[Gad96] Fabio Gadducci. On the algebraic approach to concurrent term rewriting. *Bulletin-European Association for Theoretical Computer Science*, 59:412–413, 1996.

[Jac18] Bart Jacobs. A channel-based exact inference algorithm for Bayesian networks. *arXiv preprint arXiv:1804.08032*, 2018.

[JKM+15] Ravi Jhawar, Barbara Kordy, Sjouke Mauw, Saša Radomirović, and Rolando Trujillo-Rasua. Attack trees with sequential conjunction. In *IFIP International Information Security and Privacy Conference*, pages 339–353. Springer, 2015.

[KLM03] Bernhard Kaiser, Peter Liggesmeyer, and Oliver Mäckel. A new component concept for fault trees. In *Proceedings of the 8th Australian workshop on Safety critical systems and software-Volume 33*, pages 37–46, 2003.

[KMRS10] Barbara Kordy, Sjouke Mauw, Saša Radomirović, and Patrick Schweitzer. Foundations of attack–defense trees. In *International Workshop on Formal Aspects in Security and Trust*, pages 80–95. Springer, 2010.

[KPCS14] Barbara Kordy, Ludovic Piètre-Cambacédès, and Patrick Schweitzer. DAG-based attack and defense modeling: Don't miss the forest for the attack trees. *Computer science review*, 13:1–38, 2014.

[KS07] Parvaiz Ahmed Khand and Poong Hyun Seong. An attack model development process for the cyber security of safety related nuclear digital I&C systems. In *Proceedings of the Korean Nucleary Society (KNS) Fall meeting*, 2007.

[KS17] Rajesh Kumar and Mariëlle Stoelinga. Quantitative security and safety analysis with attack-fault trees. In *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, pages 25–32. IEEE, 2017.

[LZ24] Milan Lopuhaä-Zwakenberg. Attack tree metrics are operad algebras. In *2024 IEEE 37th Computer Security Foundations Symposium (CSF)*, pages 665–679. IEEE, 2024.

[LZBS22] Milan Lopuhaä-Zwakenberg, Carlos E Budde, and Mariëlle Stoelinga. Efficient and generic algorithms for quantitative attack tree analysis. *IEEE Transactions on Dependable and Secure Computing*, 20(5):4169–4187, 2022.

[MBFB06] Miles A McQueen, Wayne F Boyer, Mark A Flynn, and George A Beitel. Time-to-compromise model for cyber risk reduction estimation. In *Quality of Protection: Security Measurements and Metrics*, pages 49–64. Springer, 2006.

[MO05] Sjouke Mauw and Martijn Oostdijk. Foundations of attack trees. In *International Conference on Information Security and Cryptology*, pages 186–198. Springer, 2005.

[PZ25] Robin Piedeleu and Fabio Zanasi. *An Introduction to String Diagrams for Computer Scientists*. Cambridge University Press, 2025.

[RS15] Enno Ruijters and Mariëlle Stoelinga. Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools. *Computer science review*, 15:29–62, 2015.

[Sch99] Bruce Schneier. Attack trees. *Dr. Dobb's journal*, 24(12):21–29, 1999.

APPENDIX

A. Proof of Theorem V.14

By the uniqueness part of Theorem IV.9, it is sufficient to prove the following two statements.

1) The family of maps
$$M^{(-)} : \mathsf{TermGraphs}_{\mathsf{AT}(B)(i,j)} \to \mathsf{BoolStoch}_R(i,j),$$
$$T \mapsto \left(M^T_{yx}\right)_{y\in 2^j,\, x\in 2^i},$$
that assign to each attack tree component $T : i \to j$ the matrix M^T defined in Theorem V.14, is a functor of channel categories.
2) The desired identity,
$$[\![T]\!]_{\mathcal{I}^{\alpha}_{\mathsf{stoch}}} = M^T, \tag{11}$$
holds for every T of the form $T = \langle f\rangle$, where $f \in \mathsf{AT}(B)$ is any function symbol.

Step 1): We need to verify that $M^{(-)}$ preserves parallel and sequential compositions, and that it preserves the basic wiring channels: the identity, copy, delete, and swap gates.

Parallel compositions: Let $T_1 : i \to j$, $T_2 : k \to l$ be attack tree components over B. For every attack $a \in \mathsf{Attacks}(T_1 \otimes T_2)$, every $(x,z) \in 2^{i+k}$ and every $(y,w) \in 2^{j+l}$, $S_{T_1\otimes T_2}(a,(x,z)) = (y,w)$ if and only if $S_{T_1}(a \circ \mathsf{inl}, x) = y$ and $S_{T_2}(a \circ \mathsf{inr}, z) = w$, where $\mathsf{inl} : \mathsf{BAS}(T_1) \to \mathsf{BAS}(T_1 \otimes T_2)$ and $\mathsf{inl} : \mathsf{BAS}(T_2) \to \mathsf{BAS}(T_1 \otimes T_2)$ are the inclusion maps from the basic attack steps of T_1 and T_2 to the basic attack steps of their parallel compositions. Moreover, we have that
$$\prod_{\substack{v\in \mathsf{BAS}(T_1\otimes T_2)\\ a(v)=i}} \overline{\alpha}_i(v) = \left(\prod_{\substack{v\in \mathsf{BAS}(T_1)\\ (a\circ \mathsf{inl})(v)=i}} \overline{\alpha}_i(v)\right) \cdot \left(\prod_{\substack{v\in \mathsf{BAS}(T_2)\\ (a\circ \mathsf{inr})(v)=i}} \overline{\alpha}_i(v)\right),$$
for all $i \in \{0,1\}$. Therefore, may split the sum in the formula defining $M^{T_1\otimes T_2}$ and redistribute the products therein to obtain:
$$M^{T_1\otimes T_2}_{(y,x),(w,z)} = M^{T_1}_{(y,x)} \cdot M^{T_2}_{(w,z)} = M^{T_1} \otimes M^{T_2}.$$

Sequential compositions: Let $T_1 : i \to j$, $T_2 : j \to k$ be attack tree components over B. For every attack $a \in \mathsf{Attacks}(T_2 \circ T_1)$, and every $x \in 2^i$, $z \in 2^k$, $S_{T_2\circ T_1}(a,x) = z$ if and only if there exists a $y \in 2^j$ such that $S_{T_1}(a \circ \mathsf{inr}, x) = y$ and $S_{T_2}(a \circ \mathsf{inl}, y) = z$, where, similar to before, inl and inr are the inclusion maps from T_1 and T_2 to $T_2 \circ T_1$. To simplify notation, write
$$\Pi(a) := \left(\prod_{a(v)=1} \overline{\alpha}_1(v)\right) \cdot \left(\prod_{a(v)=0} \overline{\alpha}_0(v)\right),$$
for any attack tree component T and attack $a \in \mathsf{Attacks}(T)$. Similar to the case of parallel compositions, we then have
$$\Pi(a) = \Pi(a \circ \mathsf{inl}) \cdot \Pi(a \circ \mathsf{inr}).$$
Putting all of these observations together, we calculate
$$\begin{aligned} M^{T_2\circ T_1}_{zx} &= \sum_{S_{T_2\circ T_1}(a,x)=z} \Pi(a) \\ &= \sum_{y\in 2^j} \sum_{S_{T_1}(a_1,x)=y} \sum_{S_{T_2}(a_2,y)=z} \Pi(a_1)\Pi(a_2) \\ &= \sum_{y\in 2^j} \left(\sum_{S_{T_1}(a_1,x)=y} \Pi(a_1)\right)\left(\sum_{S_{T_2}(a_2,y)=z} \Pi(a_2)\right) \\ &= \sum_{y\in 2^j} M^{T_1}_{yx} M^{T_2}_{zy} \\ &= \left(M^{T_2} M^{T_1}\right)_{zx}. \end{aligned}$$

Basic wiring channels: Observe that if T does not contain any basic attack step, then the products over basic attack steps in the definition of M^T are empty, and hence all summands appearing therein are 1. Moreover, when there are no basic attack steps, there is only one attack (the empty attack), and therefore, there is also at most one summand. Hence, in this case, M^T_{yx} is 1 if and only if $S_T(!\,, x) = y$ (where $! : \emptyset \to \{0,1\}$ is the empty attack). Since none of the basic wiring channels contain any basic attack steps, we can use this observation to see that they satisfy Equation (11) by

direct comparison.

Step 2): It remains to show that Equation (11) holds for all atomic attack tree components associated to the function symbols in the signature $\mathsf{AT}(B)$. For the gates $\mathtt{AND}_i$ and $\mathtt{OR}_i$, as they do not contain any basic attack steps, this follows using the same observation as for the basic wiring channels. Finally, let $b \in B$ be a basic attack step label. Then $M^{\langle b \rangle}$ is a $2^1 \times 2^0$-matrix, which we can identify with a vector $\left(M_y^{\langle b \rangle}\right)$ in R^2. Since for each $y \in \{0,1\}$, there is only one attack a that satisfies $S_{\langle b \rangle}(a) = y$, the sum defining $M^{\langle b \rangle}$ reduces to

$$M_y^{\langle b \rangle} = \begin{pmatrix} \overline{\alpha}_0(v) & \overline{\alpha}_1(v) \end{pmatrix}^{\mathsf{T}},$$

where $v \in \mathsf{BAS}(\langle b \rangle)$ is the unique basic attack step in $\langle b \rangle$. Using that $\overline{\alpha}_i(v) = \alpha_i(\mathsf{label}^T(v)) = \alpha_i(b)$ $(i \in \{0,1\})$, we obtain

$$M_y^{\langle b \rangle} = \begin{pmatrix} \alpha_0(b) & \alpha_1(b) \end{pmatrix}^{\mathsf{T}} = [\![\langle b \rangle]\!]_{\mathcal{I}^\alpha_{\mathsf{stoch}}},$$

thus completing the proof. □

B. Proof of Theorem V.21

By Theorem V.20,

$$\left([\![T]\!]_{\mathcal{I}^\alpha_{\mathsf{prop}}}\right)_1 = \left(\bigcup_{a \in \mathsf{MinSuc}_T} \prod_{a(b)=1} \overline{\alpha}(b) \right)^{\mathsf{AC}}, \tag{12}$$

where the product is taken in the semiring $\mathsf{AC}(T)$. As before, we identify attacks $a \in \mathsf{MinSuc}_T$ with subsets of B, sometimes writing $b \in a$ instead of $a(b) = 1$. Then, using the definition of the product in $\mathsf{AC}(T)$, we obtain:

$$\begin{aligned} \prod_{a(b)=1} \overline{\alpha}(b) &= \left\{ \bigcup_{b \in a} \tilde{a}_b \;\middle|\; \forall b \in a : \; \tilde{a}_b \in \overline{\alpha}(b) \right\}^{\mathsf{AC}} \\ &= \left\{ \bigcup_{b \in a} \tilde{a}_b \;\middle|\; \forall b \in a : \; \tilde{a}_b \in \{\{b\}\} \right\}^{\mathsf{AC}} \\ &= \left\{ \bigcup_{b \in a} \{b\} \right\}^{\mathsf{AC}} = \{a\}^{\mathsf{AC}} = \{a\}. \end{aligned}$$

Therefore, the union over minimal successful attacks on the right-hand side of Equation (12) is a union of singleton sets, giving,

$$\left([\![T]\!]_{\mathcal{I}^\alpha_{\mathsf{prop}}}\right)_1 = (\mathsf{MinSuc}_T)^{\mathsf{AC}} = \mathsf{MinSuc}_T,$$

where the final equality follows because MinSuc_T is already an antichain. □