Logical Methods in Computer Science
Volume 22, Issue 1, 2026, pp. 15:1–15:31
https://lmcs.episciences.org/

Submitted Dec. 23, 2024
Published Feb. 27, 2026

TERMINATION OF GRAPH TRANSFORMATION SYSTEMS VIA GENERALIZED WEIGHTED TYPE GRAPHS

JÖRG ENDRULLIS AND ROY OVERBEEK

Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
e-mail address: j.endrullis@vu.nl, overbeek.research@outlook.com

Abstract. We refine the weighted type graph technique for proving termination of double pushout (DPO) graph transformation systems. We increase the power of the approach for graphs, we generalize the technique to other categories, and we allow for variations of DPO that occur in the literature.

1. Introduction

Termination is a central aspect of program correctness. There has been extensive research on termination [Bey22, LJB01, CPR06, CPR11, AP93, GRS+11] of imperative, functional, and logic programs. Moreover, there exists a wealth of powerful termination techniques for term rewriting systems [Zan03]. Term rewriting has proven useful as an intermediary formalism to reason about C, Java, Haskell, and Prolog programs [GRS+19, GBE+14].

There are limitations to the adequacy of terms, however. Terms are usually crude representations of program states, and termination might be lost in translation. This holds especially for programs that operate with graph structures, and for programs that make use of non-trivial pointer or reference structures.

Nonetheless, such programs can be elegantly represented by graphs. This motivates the use of graph transformation systems as a unifying formalism for reasoning about programs. Correspondingly, there is a need for automatable techniques for reasoning about graph transformation systems, in particular for (dis)proving properties such as termination and confluence.

Unfortunately, there are not many techniques for proving termination of graph transformation systems. A major obstacle is the large variety of graph-like structures. Indeed, the termination techniques that do exist are usually defined for rather specific notions of graphs. This is in stark contrast to the general philosophy of algebraic graph transformation [EEPT06], the predominant approach to graph transformation, which uses the language of category theory to define and study graph transformations in a graph-agnostic manner.

In this paper, we propose a powerful graph-agnostic technique for proving termination of graph transformation systems based on weighted type graphs. We base ourselves on work by Bruggink et al. [BKNZ15]. Their method was developed specifically for edge-labelled multigraphs, using double pushout (DPO) graph transformation, and the graph rewrite rules

Key words and phrases: Graph rewriting, Termination, Weighted Type Graphs, DPO.

DOI:10.46298/LMCS-22(1:15)2026

were required to have a discrete interface, i.e., an interface without edges. We generalize and improve their approach in several ways:

(a) We strengthen the weighted type graphs technique for graph rewriting with monic matching. In the DPO literature, matches are usually required to be monic since this increases expressivity [HMP01]. The method of Bruggink et al. [BKNZ15] is not sensitive to such constraints. It always proves the stronger property of termination with respect to unrestricted matching. Consequently, it is bound to fail whenever termination depends on monic matching. Our method, by contrast, is sensitive to such constraints.
(b) We generalize weighted type graphs to arbitrary categories, and we propose the notion of 'traceable (sub)objects' as a generalization of nodes and edges to arbitrary categories.
(c) We formulate our technique on an abstract level, by describing minimal assumptions on the behaviour of pushouts involved in the rewrite steps. This makes our technique applicable to the many variants of DPO, such as those that restrict to specific pushout complements (see Remark 3.2).

Outline. In Section 2, we give a gentle introduction to our termination technique. Section 3 contains preliminaries on rewriting, termination, and semirings. We introduce a notion of weighted type graphs for general categories in Section 4. We then investigate traceability and monicity criteria that enable us to decompose the weight of pushout objects in Section 5. In Section 6, we propose a technique for proving termination of graph transformation systems. We demonstrate our technique on a number of examples in Section 7. In Section 8, we provide some benchmarks from a Scala implementation. We conclude with a discussion of related work in Section 9, followed by directions for future research in Section 10.

2. Overview of the Technique

In this section, we give a gentle introduction to our termination technique and explain the key properties that we employ to weigh pushouts.

The general idea behind our technique is as follows. DPO rewrite systems induce rewrite steps $G \Rightarrow H$ on objects if there exists a diagram of the form

$$\begin{array}{ccccc} L & \leftarrow l - & K & - r \rightarrow & R \\ \downarrow{\scriptstyle m} & \text{PO} & \downarrow{\scriptstyle u} & \text{PO} & \downarrow{\scriptstyle w} \\ G & \leftarrow l' - & C & - r' \rightarrow & H \end{array} \tag{2.1}$$

The top span of the diagram is a rule of the rewrite system. We define a decreasing measure $\mathbf{w}$ on objects, and show that $\mathbf{w}(G) > \mathbf{w}(H)$ for any such step generated by the system. We want to do this in a general categorical setting. This means that we cannot make assumptions about what the objects and morphisms represent. For simplicity, assume for now that the system contains a single rule $\rho : L \leftarrow K \rightarrow R$.

A preliminary measure $\mathbf{w}$ is $\mathbf{w} = \#\mathrm{Hom}(\,\cdot\,, T)$ for some distinguished object T. Given such a T, a termination proof then consists of finding bounds B_L and B_R such that for all steps $G \Rightarrow H$ generated by ρ,

$$\#\mathrm{Hom}(G, T) \geqslant B_L > B_R \geqslant \#\mathrm{Hom}(H, T)\ .$$

These bounds would have to be constructed using only ρ, T, and generic facts and assumptions about pushout squares. The fact that the two pushout squares of a DPO step share the

vertical morphism $u : K \to C$ is useful here, because naively, $G = L \cup_K C$ and $H = R \cup_K C$, meaning that K and C are factors common to G and H.

In the remainder of this section, we refine the preliminary measure into something more powerful (Section 2.1), explain which bounds we want to consider (Section 2.2), and explain how to construct these bounds from pushout properties (Section 2.3).

2.1. **Weighing morphisms into a type graph.** We equip T with a **weighted set of T-valued elements** $(\mathbb{E}, \mathbf{w})$, where $\mathbb{E}$ contains morphisms e with codomain T, and $w : \mathbb{E} \to \mathcal{S}$ is a weight assignment into a well-founded commutative semiring $\mathcal{S}$. This gives rise to a notion of **weighted type graphs** (see Definition 4.1).

Morphisms $\phi : G \to T$ are assigned a weight depending on how they relate to these weighted elements. A priori there are different ways of doing this, but we will count for each $e \in \mathbb{E}$ the number $n(e)$ of commuting triangles of the form

$$\begin{array}{ccc} dom(e) & \xrightarrow{\alpha} & G \\ & \searrow^{e} \quad = \quad \swarrow_{\phi} & \\ & T & \end{array} \tag{2.2}$$

We then compute the **weight of a morphism** $\phi : G \to T$ by

$$\mathbf{w}(\phi) = \prod_{e \in \mathbb{E}} \mathbf{w}(e)^{n(e)} \tag{2.3}$$

Finally, the **weight of an object** G is defined as the sum of the weights of the morphisms in $\mathrm{Hom}(G, T)$ (see further Definition 4.4).

2.2. **Lower and upper bounds for pushouts.** Our goal is to establish that the rewrite step is decreasing, that is, $\mathbf{w}(G) > \mathbf{w}(H)$, by considering only the rule $\rho : L \leftarrow K \to R$. For this purpose, we aim to decompose the weights of G and H as follows:

$$\mathbf{w}(G) \;\geq\; \mathbf{w}(C) \cdot \mathbf{w}(L) \qquad\qquad \mathbf{w}(C) \cdot \mathbf{w}(R) \;\geq\; \mathbf{w}(H) \tag{2.4}$$

The decomposition gives us a lower bound on $\mathbf{w}(G)$ and an upper bound on $\mathbf{w}(H)$. Due to the common factor $\mathbf{w}(C)$, for proving $\mathbf{w}(G) > \mathbf{w}(H)$, it suffices to show that $\mathbf{w}(L) > \mathbf{w}(R)$. Thus, we only need to analyze the rule and not all rewrite steps (see Definition 6.9).

Although this gives the general idea, the bounds of Equation (2.4) actually overestimate the weight. The weight of (an image of) the interface $\mathbf{w}(K)$ appears as a part of $\mathbf{w}(C)$, $\mathbf{w}(L)$, and $\mathbf{w}(R)$. As a consequence, $\mathbf{w}(K)$ is counted twice in $\mathbf{w}(C) \cdot \mathbf{w}(L)$ and twice in $\mathbf{w}(C) \cdot \mathbf{w}(R)$. To avoid this overcounting, we improve the bounds as follows:

$$\mathbf{w}(G) \;\geq\; \mathbf{w}(C - u) \cdot \mathbf{w}(L) \qquad\qquad \mathbf{w}(C - u) \cdot \mathbf{w}(R) \;\geq\; \mathbf{w}(H) \tag{2.5}$$

Here, $\mathbf{w}(C - u)$ is the weight of C excluding those morphisms $e : dom(e) \to C$, for $e \in \mathbb{E}$, that factor through $u : K \to C$ (see further Definition 4.4).

2.3. **Assumptions about pushout squares.** Finally, we need to identify assumptions about pushout squares that allow us to decompose the weights of G and H as in Equation (2.5). For simplicity, assume that T has a single weighted element $e : X \to T$.

First, consider the right pushout square of the rewrite step together with a morphism $\alpha : X \to H$. The morphism α could potentially give rise to a commuting triangle

$$\begin{array}{ccc} X & \xrightarrow{\alpha} & H \\ & {\scriptstyle e} \searrow \quad = \quad \swarrow {\scriptstyle \phi} & \\ & T & \end{array} \quad , \tag{2.6}$$

and thereby contribute to the weight $\mathbf{w}(H)$ of H. For $\mathbf{w}(C-u)\cdot\mathbf{w}(R)$ to be an upper bound for $\mathbf{w}(H)$, the morphism α needs to be **traceable** back to C or R. This means that, in the following diagram, at least one of the dotted morphisms must exist and form a commuting triangle (see further Definition 5.2):

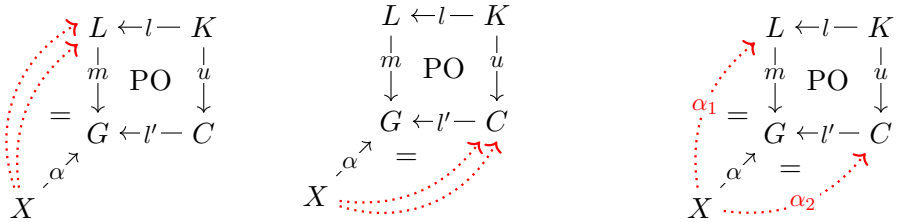
$$\begin{array}{ccccc} L & \xleftarrow{l} & K & \xrightarrow{r} & R \\ \downarrow{\scriptstyle m} & \text{PO} & \downarrow{\scriptstyle u} & \text{PO} & \downarrow{\scriptstyle w} \\ G & \xleftarrow{l'} & C & \xrightarrow{r'} & H \end{array} \quad \begin{array}{l} = \\ = \\ = \end{array} \quad \begin{array}{c} \\ \\ \nwarrow{\scriptstyle \alpha} \\ X \end{array} \tag{2.7}$$

For general pushouts in **Graph**, for instance, any node or edge in the pushout object H traces back to a node or edge in R or C – the same is not true for loops in H, which can be created by the pushout. For pushouts along monomorphisms in **Graph**, however, these "traceable elements" do include loops. This highlights that traceability depends on

(a) what elements we are tracing (namely, $dom(e)$ for $e \in \mathbb{E}$), as well as
(b) restrictions on the pushout squares in rewrite steps (e.g., monic r in the rule).

We will show that traceability suffices to obtain the upper bound on $\mathbf{w}(H)$ (Definition 5.9 and Lemma 5.10).

Second, consider the left pushout square of the rewrite step together with a morphism $\alpha : X \to G$. For $\mathbf{w}(C-u)\cdot\mathbf{w}(L)$ to be a lower bound for $\mathbf{w}(G)$, we must avoid overcounting in $\mathbf{w}(C-u)$ and $\mathbf{w}(L)$. The problem of overcounting occurs whenever α traces back to more than one morphism in L or $C-u$. There are three possible scenarios, as shown in Figure 1.



(a) Overcounting in L. (b) Overcounting in C. (c) Overcounting between L and C.

FIGURE 1. Causes of $\mathbf{w}(C-u)\cdot\mathbf{w}(L)$ overestimating the weight $\mathbf{w}(G)$ of G.

For the dotted morphisms with codomain C, we are interested only in those that do not factor through u. The morphisms that factor through u do not contribute to overcounting

since we exclude them in $\mathbf{w}(C - u)$. The three causes of overcounting can be avoided by using the following assumptions:

(a) Overcounting in L can be prevented by requiring the match morphism m to be X-monic, that is, monic for morphisms with domain X (see Definition 5.7). For instance, think of X as an edge and of m as preserving distinctness of edges.
(b) Overcounting in C can be prevented by requiring the morphism l' to be X-monic outside of u, that is, monic for those morphisms with domain X that do not factor through u (see Definition 5.7).
(c) Overcounting between L and C can be excluded by requiring vertical strong traceability (for the left pushout squares of rewrite steps), that is, the morphism $\alpha_2 : X \to C$ must trace back to the interface K as shown in Figure 2 (see Definition 5.2). Then α_2 factors through u and is thus not counted in $\mathbf{w}(C - u)$.

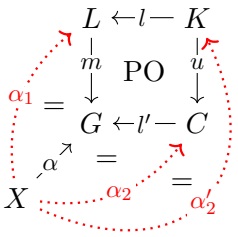


FIGURE 2. Vertical strong traceability.

Given the above assumptions, the weight of G can be decomposed as $\mathbf{w}(G) = \mathbf{w}(C-u)\cdot\mathbf{w}(L)$ (see further Definition 5.9 and Lemma 5.10).

Remainder of the proof. In Section 5, we investigate the properties of (strong) traceability in Definition 5.2 and relative monicity in Definition 5.7. We show how to weigh morphisms rooted in pushout objects (Lemma 5.10), and then extend this to weighing pushout objects (Lemma 5.13). In Section 6, we analyze termination of DPO graph transformation systems. We propose the concept of context-closures (Definition 6.5) to ensure that every rewritable object (not in normal form) admits some morphism into the type graph T. We then introduce decreasing rules (Definition 6.9) and show that these give rise to decreasing steps (Theorem 6.11). This paves the way for our termination technique (Theorem 6.12).

Remark 2.1 (Adhesivity)**.** We note that the termination technique discussed in this paper does not require general categorical properties like $\mathcal{M}$-adhesivity [EGH10, Definition 2.4]. The properties that are crucial for our technique, such as (strong) traceability and X-monicity, do not depend on the category only. They depend on a combination of

(a) the category,
(b) the elements being traced/weighed,
(c) the rewrite rules, and
(d) the double pushout framework (restrictions on the pushout squares in rewrite steps).

However, $\mathcal{M}$-adhesion can be used to establish strong traceability; see the paragraph before Proposition 5.3.

3. Preliminaries

We assume an understanding of basic category theory, in particular of pushouts, pullbacks, and monomorphisms $\rightarrowtail$. Throughout this paper, we work in a fixed category $\mathbf{C}$. For an introduction to DPO graph transformation, see [KNPR18, EPS73, CMR+97].

We write **Graph** for the category of *finite* multigraphs, and **SGraph** for the category of *finite* simple graphs (i.e., without parallel edges). The graphs can be edge-labelled and/or node-labelled over a fixed label set.

Definition 3.1 (DPO Rewriting [EPS73])**.** A **DPO rewrite rule** ρ is a span $L \leftarrow l - K - r \rightarrow R$. A diagram of the form

$$\begin{array}{ccccc} L & \leftarrow l - & K & - r \rightarrow & R \\ \downarrow m & \text{PO} & \downarrow & \text{PO} & \downarrow \\ G & \longleftarrow & C & \longrightarrow & H \end{array}$$

defines a **DPO rewrite step** $G \Rightarrow^{\rho,m}_{\text{DPO}} H$, i.e., a step from G to H using rule ρ and match morphism $m : L \to G$.

Remark 3.2. Usually Definition 3.1 is accompanied with constraints. Most commonly, l and m are required to be (regular) monic. Sometimes also the left pushout square is restricted, e.g., it may be required that the pushout complement is **minimal** or **initial** [BGS11, BHK22, BHK21]. Because such restrictions affect whether a DPO rewrite system is terminating, we will define our approach in a way that is parametric in the variation of DPO under consideration (see Definition 6.2).

In this paper, we investigate techniques for proving **strong** normalization (termination), meaning that every rewrite sequence eventually ends in a normal form (a term that can no longer be rewritten). This is in contrast to the concept of **weak** normalization which only requires the existence of some rewrite sequence to a normal form.

Definition 3.3 (Termination)**.** Let R, S be binary relations on $Ob(\mathbf{C})$. Relation R is **terminating (or strongly normalizing) relative to** S, denoted $\text{SN}(R/S)$, if every (finite or infinite) sequence

$$o_1 \ (R \cup S) \ o_2 \ (R \cup S) \ o_3 \ (R \cup S) \ o_4 \ldots \qquad \text{where } o_1, o_2, \ldots \in Ob(\mathbf{C})$$

contains only a finite number of R steps. The relation R is **terminating (or strongly normalizing)**, denoted $\text{SN}(R)$, if R terminates relative to the empty relation $\varnothing$.

These concepts carry over to sets of rules via the induced rewrite relations.

We will interpret weights into a well-founded semiring as defined below.

Definition 3.4. A **monoid** $\langle\, S, \cdot, e \,\rangle$ is a set S with an operation $\cdot : S \times S \to S$ and an **identity element** e such that for all $a, b, c \in S$:

$$(a \cdot b) \cdot c = a \cdot (b \cdot c) \qquad \text{and} \qquad a \cdot e = a = e \cdot a$$

The monoid is **commutative** if for all $a, b \in S$: $a \cdot b = b \cdot a$.

Definition 3.5. A **semiring** is a tuple $\mathcal{S} = \langle\, S, \oplus, \odot, \mathbf{0}, \mathbf{1} \,\rangle$ where S is a set and all of the following conditions hold:

(i) $\langle\, S, \oplus, \mathbf{0} \,\rangle$ is a commutative monoid;

(ii) $\langle S, \odot, \mathbf{1} \rangle$ is a monoid;
(iii) $\mathbf{0}$ is an annihilator for $\odot$: for all $a \in S$ we have

$$\mathbf{0} \odot a = \mathbf{0} = a \odot \mathbf{0} \qquad \text{(annihilation)}$$

(iv) $\odot$ distributes over $\oplus$: for all $a, b, x \in S$ we have

$$(a \oplus b) \odot x = (a \odot x) \oplus (b \odot x) \qquad \text{(right-distributivity)}$$
$$x \odot (a \oplus b) = (x \odot a) \oplus (x \odot b) \qquad \text{(left-distributivity)}$$

The semiring $\mathcal{S}$ is **commutative** if $\langle S, \odot, \mathbf{1} \rangle$ is commutative. We will often denote a semiring by its carrier set, writing S for $\mathcal{S}$.

Definition 3.6. A **well-founded semiring** $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1}, \prec, \leq \rangle$ consists of

(i) a semiring $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1} \rangle$; and
(ii) non-empty orders $\prec, \leq \subseteq S \times S$ for which $\prec \subseteq \leq$ and $\leq$ is reflexive;

such that $\mathrm{SN}(\succ/\geq)$, $\mathbf{0} \neq \mathbf{1}$, and for all $x, x', y, y' \in S$ we have

$$x \leq x' \ \wedge \ y \leq y' \quad \Longrightarrow \quad x \oplus y \ \leq \ x' \oplus y' \tag{S1}$$
$$x \prec x' \ \wedge \ y \prec y' \quad \Longrightarrow \quad x \oplus y \ \prec \ x' \oplus y' \tag{S2}$$
$$x \leq x' \ \wedge \ \mathbf{1} \leq y \quad \Longrightarrow \quad x \odot y \ \leq \ x' \odot y \ \text{ and } \ y \odot x \ \leq \ y \odot x' \tag{S3}$$
$$x \prec x' \ \wedge \ \mathbf{1} \leq y \neq \mathbf{0} \quad \Longrightarrow \quad x \odot y \ \prec \ x' \odot y \ \text{ and } \ y \odot x \ \prec \ y \odot x' \tag{S4}$$

The semiring is **strictly monotonic** if additionally

$$x \prec x' \ \wedge \ y \leq y' \quad \Longrightarrow \quad x \oplus y \ \prec \ x' \oplus y' \tag{S5}$$

Proposition 3.7. *Let $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1}, \prec, \leq \rangle$ be a well-founded semiring, then*

$$\mathbf{1} \leq x, y \ \wedge \ x, y \neq \mathbf{0} \quad \Longrightarrow \quad \mathbf{1} \ \leq \ (x \odot y) \ \neq \ \mathbf{0} \tag{S6}$$
$$\mathbf{1} \leq a, b \ \wedge \ x \leq y \quad \Longrightarrow \quad (a \oplus b) \odot x \ \leq \ (a \oplus b) \odot y \tag{S7}$$
$$\mathbf{1} \leq a, b \ \wedge \ a, b \neq \mathbf{0} \ \wedge \ x \prec y \quad \Longrightarrow \quad (a \oplus b) \odot x \ \prec \ (a \oplus b) \odot y \tag{S8}$$

for all $x, y, a, b \in S$.

Proof. For (S6), assume that $\mathbf{1} \leq x, y$ and $x, y \neq \mathbf{0}$. We get $\mathbf{1} \leq y = \mathbf{1} \odot y \leq x \odot y$ from (S3). By definition, there exist $c, d \in S$ with $c \prec d$, and hence $x \odot y = \mathbf{0}$ would contradict $c \odot x \odot y \prec d \odot x \odot y$ by (S4).

For (S7), let $\mathbf{1} \leq a, b$ and $x \leq y$. We have

$$(a \oplus b) \odot x = (a \odot x) \oplus (b \odot x) \leq (a \odot y) \oplus (b \odot y) = (a \oplus b) \odot y$$

using distributivity and (S2) since $a \odot x \leq a \odot y$ and $b \odot x \leq b \odot y$ by (S3).

For (S8), let $\mathbf{1} \leq a, b$, $a, b \neq \mathbf{0}$ and $x \prec y$. We have

$$(a \oplus b) \odot x = (a \odot x) \oplus (b \odot x) \prec (a \odot y) \oplus (b \odot y) = (a \oplus b) \odot y$$

using distributivity and (S2) since $a \odot x \prec a \odot y$ and $b \odot x \prec b \odot y$ by (S4). □

Example 3.8 (Arithmetic semiring)**.** The **arithmetic semiring** $\langle \mathbb{N}, \oplus, \odot, \mathbf{0}, \mathbf{1}, \prec, \leq \rangle$, over the natural numbers $\mathbb{N}$, is a strictly monotonic, well-founded semiring where $\oplus = +$, $\odot = \cdot$, $\mathbf{0} = 0$, $\mathbf{1} = 1$, and $\prec = <$ and $\leq$ are the usual orders on $\mathbb{N}$.

Example 3.9 (Tropical semiring)**.** The **tropical semiring** $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1} \rangle$ has carrier set $S = \mathbb{N} \cup \{\infty\}$. The usual order $\leq$ on $\mathbb{N}$ is extended to S by letting $a \leq \infty$ for all $a \in S$. The operation $+$ on $\mathbb{N}$ is extended to S by $a + \infty = \infty$ for all $a \in S$. Then the operations and unit elements of the tropical semiring are given by:

$$\oplus(a, b) = \min\{ a, b \} \qquad \odot(a, b) = a + b \qquad \mathbf{0} = \infty \qquad \mathbf{1} = 0$$

The tropical semiring is a well-founded semiring $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1}, \prec, \leq \rangle$, where $\prec$ is the strict part of $\leq$. It is not strictly monotonic because $2 \oplus 2 \not\prec 3 \oplus 2$.

Example 3.10 (Arctic semiring)**.** The **arctic semiring** $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1} \rangle$ has carrier set $S = \mathbb{N} \cup \{-\infty\}$. The usual order $\leq$ on $\mathbb{N}$ is extended to S by $-\infty \leq a$ for all $a \in S$. The operation $+$ on $\mathbb{N}$ is extended to S by $a + (-\infty) = -\infty$ for all $a \in S$. The operations and unit elements of the arctic semiring are:

$$\oplus(a, b) = \max\{ a, b \} \qquad \odot(a, b) = a + b \qquad \mathbf{0} = -\infty \qquad \mathbf{1} = 0$$

The arctic semiring is a well-founded semiring $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1}, \prec, \leq \rangle$ where $\prec$ is the strict part of $\leq$. It is not strictly monotonic because $2 \oplus 3 \not\prec 3 \oplus 3$.

4. Weighted Type Graphs

In this section, we introduce a notion of weighted type graphs for arbitrary categories, and we employ them to weigh morphisms (into the type graph) and objects in the category.

Definition 4.1 (Weighted type graphs)**.** A **weighted type graph** $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ consists of:

(i) an object $T \in Ob(\mathbf{C})$, called a **type graph**,
(ii) a set $\mathbb{E}$ of T**-valued elements** $e : dom(e) \to T$,
(iii) a commutative semiring $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1} \rangle$, and
(iv) a **weight function** $\mathbf{w} : \mathbb{E} \to S$ such that, for all $e \in \mathbb{E}$, $\mathbf{0} \neq \mathbf{w}(e) \geq \mathbf{1}$.

The weighted type graph $\mathcal{T}$ is **finitary** if, for every $e \in \mathbb{E}$ and every $G \in Ob(\mathbf{C})$, $\mathrm{Hom}(dom(e), G)$ and $\mathrm{Hom}(G, T)$ are finite sets.

Remark 4.2. Bruggink et al. [BKNZ15] introduce weighted type graphs for the category **Graph** of edge-labelled multigraphs. We generalize this concept to arbitrary categories. Important technical issues aside (discussed below), their notion is obtained by instantiating our notion of weighted type graphs for **Graph** with $\mathbb{E} = \bigcup \{ \mathrm{Hom}(\bullet \xrightarrow{x} \bullet, T) \mid x \text{ an edge label} \}$.

The technical differences are as follows. We require $\mathbf{w}(e) \geq \mathbf{1}$ and $\mathbf{w}(e) \neq \mathbf{0}$ for every $e \in \mathbb{E}$. The work [BKNZ15] requires

- $\mathbf{w}(e) > \mathbf{0}$ only for loops $e \in \mathbb{E}$ on a distinguished 'flower node'.

This condition is insufficient, causing [BKNZ15, Theorem 2] to fail as illustrated by [EO24a, Example A.1]. In the revised version [BKNZ23], this problem has been fixed.

Moreover, the condition $\mathbf{w}(e) > \mathbf{0}$ rules out the use of the tropical semiring as $\mathbf{0} = +\infty$. Observe that $\mathbf{w}(e) \geq \mathbf{1}$ is often a weaker condition than $\mathbf{w}(e) > \mathbf{0}$. In particular, our condition enables the use of the tropical semiring (for instance, see Example 7.4).

Notation 4.3. *We use the following notation:*

(a) *For* $\alpha : A \to B$, $x : A \to C$, *let* $\{ - \circ \alpha = x \} = \{ \phi : B \to C \mid \phi \circ \alpha = x \}$;
(b) *For* $\alpha : B \to C$, $x : A \to C$, *let* $\{ \alpha \circ - = x \} = \{ \phi : A \to B \mid \alpha \circ \phi = x \}$;

(c) *For $\alpha : B \to C$ and a set S of morphisms $\phi : A \to B$, we define* $\alpha \circ S = \{ \alpha \circ \phi \mid \phi \in S \}$.

Definition 4.4 (Weighing morphisms and objects)**.** Let $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ be a finitary weighted type graph and let $G, A \in Ob(\mathbf{C})$.

(i) The **weight of a morphism** $\phi : G \to T$ is $\mathbf{w}_{\mathcal{T}}(\phi) = \bigodot_{e \in \mathbb{E}} \mathbf{w}(e)^{\#\{ \phi \circ - = e \}}$.

This is equivalent to $\mathbf{w}_{\mathcal{T}}(\phi) = \bigodot_{\substack{e \in \mathbb{E} \\ \alpha \in \{ \phi \circ - = e \}}} \mathbf{w}(e)$; visually $\begin{array}{ccc} dom(e) & \overset{\alpha}{\to} & G \\ & {\scriptstyle e} \searrow \quad = \quad \swarrow {\scriptstyle \phi} & \\ & T & \end{array}$.

(ii) The **weight of finite** $\Psi \subseteq \{ \phi \mid \phi : G \to T \}$ is $\mathbf{w}_{\mathcal{T}}(\Psi) = \bigoplus_{\phi \in \Psi} \mathbf{w}_{\mathcal{T}}(\phi)$.

(iii) The **weight of an object** $G \in Ob(\mathbf{C})$ is $\mathbf{w}_{\mathcal{T}}(G) = \mathbf{w}_{\mathcal{T}}(\mathrm{Hom}(G, T))$.

(iv) The **weight of** $\phi : G \to T$ **excluding** $(\alpha \circ -)$, for $\alpha : A \to G$, is

$$\mathbf{w}_{\mathcal{T}}(\phi - (\alpha \circ -)) = \bigodot_{e \in \mathbb{E}} \mathbf{w}(e)^{\#\{ \tau \in \{ \phi \circ - = e \} \mid \nexists \zeta.\ \tau = \alpha \circ \zeta \}}$$

In other words, we count all τ such that for all $\zeta : dom(e) \to A$, we have the following:

$$\begin{array}{ccccc} & & A & & \\ & \overset{\zeta}{\nearrow} & \neq & \overset{\alpha}{\searrow} & \\ dom(e) & & \xrightarrow{\quad \tau \quad} & & G \\ & \underset{e}{\searrow} & = & \underset{\phi}{\swarrow} & \\ & & T & & \end{array}$$

Observe the analogy to weighted automata where the weight of a word is the sum of the weights of all runs, and the weight of a run is the product of the edge weights.

Lemma 4.5. *Let $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ be a finitary weighted type graph. Then*
(1) *for every $\phi : G \to T$: $\mathbf{1} \leq \mathbf{w}_{\mathcal{T}}(\phi) \neq \mathbf{0}$;*
(2) *for every $\phi : G \to T$ and $\alpha : A \to G$: $\mathbf{1} \leq \mathbf{w}_{\mathcal{T}}(\phi - (\alpha \circ -)) \neq \mathbf{0}$;*

Proof. Follows from $\mathbf{1} \leq \mathbf{w}(e) \neq \mathbf{0}$ for every $e \in \mathbb{E}$ and Proposition 3.7. □

5. Weighing Pushouts

In this section, we introduce techniques for determining exact and upper bounds on the weights of pushout objects in terms of the corresponding pushout span. For these techniques to work, we need certain assumptions on the pushout squares. This concerns, in particular, traceability of elements (Section 5.1) and relative monicity conditions (Section 5.2), which are used to define the notion of weighable pushout squares (Section 5.3). Using the notion of a weighable pushout square, we can bound weights of morphisms out of the pushout object (Section 5.4), and consequently bound the weight of the pushout object itself (Section 5.5).

The following definition introduces an orientation for pushout squares, distinguishing between horizontal and vertical morphisms. A double pushout rewrite step with respect to a rule consists of two oriented pushout squares. For both squares, β is the shared morphism, and α will be l or r of the rule span, respectively. For the left square, β' is the match morphism. The distinction between horizontal and vertical morphisms will be used in Definitions 5.2 and 5.9 below to impose different conditions on these morphisms.

Definition 5.1 (Oriented pushout squares)**.** An **oriented pushout square** τ is a pushout square with a fixed orientation:

$$\begin{array}{ccc} A & \xrightarrow{\alpha} & B \\ \downarrow{\scriptstyle \beta} & \mathrm{PO} & \downarrow{\scriptstyle \beta'} \\ C & \xrightarrow{\alpha'} & D \end{array}$$

We distinguish the morphisms into horizontal and vertical. Morphisms β and β' are **vertical** and α and α' are **horizontal**.

We denote the pushout square by $\langle\, C \overset{\beta}{\leftarrow} A \overset{\alpha}{\rightarrow} B;\ C \overset{\alpha'}{\rightarrow} D \overset{\beta'}{\leftarrow} B \,\rangle$ where the order of the morphisms in the span is vertical before horizontal, and the order of the morphisms in the cospan is horizontal before vertical.

5.1. **Traceability.** We introduce the concept of traceability of elements $X \in Ob(\mathbf{C})$ along pushout squares. Roughly speaking, traceability guarantees that the pushout cannot create occurrences of X out of thin air.

Definition 5.2 (Traceability)**.** Let Δ be a class of oriented pushout squares. An object $X \in Ob(\mathbf{C})$ is called **traceable along** Δ if: whenever we have a configuration of the form

$$\begin{array}{ccc} A & \xrightarrow{\alpha} & B \\ \downarrow{\scriptstyle \beta} & \Delta & \downarrow{\scriptstyle \beta'} \\ C & \xrightarrow{\alpha'} & D \end{array} \qquad \begin{array}{c} \\ \\ \\ \nwarrow f \\ X \end{array},$$

such that the displayed square is from Δ, then

(a) there exists a $g : X \to B$ such that $f = \beta' g$, or
(b) there exists an $h : X \to C$ such that $f = \alpha' h$.

Moreover we say that

(c) X is **vertically strongly traceable along** Δ if:
whenever (a) and (b) hold together, then there exists $h' : X \to A$ such that $h = \beta h'$; [1][2]
(d) X is **horizontally strongly traceable along** Δ if:
whenever (a) and (b) hold together, then there exists $g' : X \to A$ such that $g = \alpha g'$;
(e) X is **strongly traceable along** Δ if:
X is both horizontally and vertically strongly traceable along Δ.

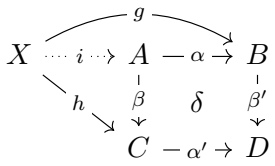
In $\mathcal{M}$-adhesive categories (for $\mathcal{M}$ a stable class of monos) [EGH10, Definition 2.4], pushouts along $\mathcal{M}$-morphisms are also pullbacks [LS04, Lemma 13]. This is a common setting for graph transformation. In this case, the notions of traceability and strong traceability coincide for pushouts along $\mathcal{M}$-morphisms, as the proposition below shows.

[1]For our termination technique we only need vertical strong traceability. However, in all our examples, strong traceability holds horizontally and vertically.

[2]In [EO24b] mistakenly a weaker condition has been stated (requiring only the existence of $f' : X \to A$ with $f = \beta' \alpha f'$). This weaker condition is insufficient as it causes a problem with the property $(\star)$ in the proof of Lemma 5.10.

Proposition 5.3. *If Δ is a class of pushouts that are also pullbacks, then traceability along Δ implies strong traceability along Δ (for $X \in Ob(\mathbf{C})$).*

Proof. If conditions (a) and (b) of Definition 5.2 hold, the outer diagram of

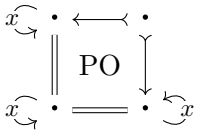


commutes, so that i is obtained (uniquely) by the pullback property of $\delta \in \Delta$, making the respective triangles commute. □

Remark 5.4 (Traceability in **Graph**)**.** In **Graph**, the objects $\bullet$ and $\bullet \xrightarrow{x} \bullet$ (for edge labels x) are the only (non-initial) objects that are (strongly) traceable along all pushout squares. Other objects, such as loops $\bullet \circlearrowleft$, are strongly traceable if all morphisms in the square are monomorphisms.

Remark 5.5 (Traceability in **SGraph**)**.** In the category of simple graphs **SGraph**, object $\bullet$ is strongly traceable along all pushout squares. An object $X = \bullet \xrightarrow{x} \bullet$ (for an edge label x) is traceable along all pushout squares, but not necessarily strongly traceable.

The following pushout is a counterexample:



However, using Proposition 5.3, we have that X is strongly traceable along pushout squares in which one of the span morphisms α or β is a regular monomorphism, because such pushouts are pullbacks in **SGraph** (and more generally in quasitoposes) [JLS07, Proposition 10 & Corollary 18].

Conjecture 5.6. As is well known, the category of unlabeled graphs is equivalent to the functor category $[\,\mathrm{E} \rightrightarrows \mathrm{V}\,, \mathbf{Set}]$, where index category $\mathrm{E} \rightrightarrows \mathrm{V}$ is the category consisting of two objects E and V and two non-identity arrows that are parallel. This makes the category of unlabeled graphs a presheaf category, and the two representable functors for this category are precisely the one node graph $\cdot$ and the one edge graph $\cdot \to \cdot$ (see [Vig03, Section 3] for more detail). We identified these two graphs as non-initial objects that are strongly traceable along any pushout (Remark 5.4). Our conjecture is that for any presheaf category for which the index category is small and acyclic, the non-initial strongly traceable objects for any pushout are precisely the representable functors. For many examples of such presheaf categories, see the overview provided by Löwe [Löw93, Section 3.1] (there called graph structures).

5.2. **Relative Monicity.** Traceability suffices to derive an upper bound on the weight of a pushout. However, we need additional monicity assumptions for lower or exact bounds.

Definition 5.7 (Relative monicity)**.** A morphism $f : A \to B$ is called

(i) **monic for** $S \subseteq \mathrm{Hom}(-, A)$ if $fg = fh \implies g = h$ for all $g, h \in S$,
(ii) X**-monic**, where $X \in Ob(\mathbf{C})$, if f is monic for $\mathrm{Hom}(X, A)$, and
(iii) X**-monic outside of** u, where $X \in Ob(\mathbf{C})$ and $u : C \to A$, if f is monic for $\mathrm{Hom}(X, A) - u \circ \mathrm{Hom}(X, C)$.

For $\Gamma \subseteq Ob(\mathbf{C})$, f is called Γ**-monic** if f is X-monic for every $X \in \Gamma$. When Γ is clear from the context, $A \diamond\!\!\rightarrow B$ denotes a Γ-monic morphism from A to B.

Example 5.8 (Edge-Monicity)**.** In **Graph**, let $\Gamma = \{\bullet \xrightarrow{x} \bullet \mid x \text{ is an edge label}\}$. A morphism $f : G \to H$ that does not identify edges (but may identify nodes) is not necessarily monic, but it is Γ-monic. In this particular case for **Graph**, we will say that f is **edge-monic**.

5.3. **Weighable Pushout Squares.** The following definition summarizes the conditions required to derive an upper or exact bound on the weight of a pushout object.

Definition 5.9 (Weighable pushout square)**.** Let $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ be a finitary weighted type graph. Consider an oriented pushout square δ:

$$\begin{array}{ccc} A & -\alpha\rightarrow & B \\ \mid & & \mid \\ \beta & \delta & \beta' \\ \downarrow & & \downarrow \\ C & -\alpha'\rightarrow & D \end{array}$$

We say that δ is

(a) **weighable** with $\mathcal{T}$ if:
  (i) $dom(\mathbb{E})$ is vertically strongly traceable along δ,
  (ii) β' is $dom(\mathbb{E})$-monic, and
  (iii) α' is $dom(\mathbb{E})$-monic outside of β.
(b) **bounded above** by $\mathcal{T}$ if $dom(\mathbb{E})$ is traceable along δ.

5.4. **Weighing Morphisms Rooted in Pushout Objects.** Consider the pushout diagram in Lemma 5.10, below. The lemma decomposes the weight $\mathbf{w}_{\mathcal{T}}(\phi)$ of the morphism ϕ, rooted in the pushout object, into the product of the weights $\mathbf{w}_{\mathcal{T}}(\phi \circ \beta')$ and $\mathbf{w}_{\mathcal{T}}(\phi \circ \alpha' - (\beta \circ -))$. Intuitively, the reasoning is as follows. Let $e \in \mathbb{E}$ be a weighted element. Assume that $e : dom(e) \to T$ occurs in ϕ, that is, there exists $d : dom(e) \to D$ with $e = \phi \circ d$. Then traceability guarantees that this occurrence d can be traced back to $\phi \circ \beta'$ or $\phi \circ \alpha'$. In case the occurrence d traces back all the way to A (to $\phi \circ \beta' \circ \alpha = \phi \circ \alpha' \circ \beta$), then it occurs in both $\phi \circ \beta'$ and $\phi \circ \alpha'$. To avoid unnecessary double counting, we exclude those elements from $\phi \circ \alpha'$, giving rise to $\mathbf{w}_{\mathcal{T}}(\phi \circ \alpha' - (\beta \circ -))$. In this way, we obtain the following upper bound on the weight of ϕ:

$$\mathbf{w}_{\mathcal{T}}(\phi) \;\leq\; \mathbf{w}_{\mathcal{T}}(\phi \circ \beta') \odot \mathbf{w}_{\mathcal{T}}(\phi \circ \alpha' - (\beta \circ -))$$

To obtain the precise weight we need to avoid double counting. First, we need vertical strong traceability to exclude all double counting between $\phi \circ \beta'$ and $\phi \circ \alpha' - (\beta \circ -)$. Second, we need $dom(\mathbb{E})$-monicity of α' outside of β and $dom(\mathbb{E})$-monicity of β' to avoid double counting along these morphisms.

Lemma 5.10 (Weighing morphisms rooted in pushout objects)**.** *Let a finitary weighted type graph $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ be given. Consider an oriented pushout square δ:*

$$\begin{array}{ccc} A & \xrightarrow{\alpha} & B \\ \downarrow{\scriptstyle \beta} & \delta & \downarrow{\scriptstyle \beta'} \\ C & \xrightarrow{\alpha'} & D \xrightarrow{\phi} T \end{array}$$

We define $k = \mathbf{w}_{\mathcal{T}}(\phi \circ \beta') \odot \mathbf{w}_{\mathcal{T}}(\phi \circ \alpha' - (\beta \circ -))$. *Then the following holds:*

(A) *We have* $\mathbf{w}_{\mathcal{T}}(\phi) = k$ *if* δ *is weighable with* $\mathcal{T}$.
(B) *We have* $\mathbf{w}_{\mathcal{T}}(\phi) \leq k$ *if* δ *is bounded above by* $\mathcal{T}$.

Proof. Recall the relevant definitions:

$$\mathbf{w}_{\mathcal{T}}(\phi) = \bigodot_{e \in \mathbb{E}} \mathbf{w}(e)^{\#\{\, \phi \circ - = e \,\}}$$
$$\mathbf{w}_{\mathcal{T}}(\phi \circ \beta') = \bigodot_{e \in \mathbb{E}} \mathbf{w}(e)^{\#\{\, \phi \circ \beta' \circ - = e \,\}}$$
$$\mathbf{w}_{\mathcal{T}}(\phi \circ \alpha' - (\beta \circ -)) = \bigodot_{e \in \mathbb{E}} \mathbf{w}(e)^{\#\{\, \tau \in \{\, \phi \circ \alpha' \circ - = e \,\} \mid \nexists \zeta.\ \tau = \beta \circ \zeta \,\}}$$

For part (A) of the lemma, it suffices to prove that

$$\begin{aligned} \#\{\, \phi \circ - = e \,\} = {} & \#\{\, \phi \circ \beta' \circ - = e \,\} + \\ & \#\{\, \tau \in \{\, \phi \circ \alpha' \circ - = e \,\} \mid \nexists \zeta.\ \tau = \beta \circ \zeta \,\} \end{aligned} \tag{5.1}$$

for every $e \in \mathbb{E}$. Moreover, for part (B) of the lemma, it suffices to prove Equation (5.1) with $\leq$ instead of $=$ (because by definition $\mathbf{1} \leq \mathbf{w}(e)$ for every $e \in \mathbb{E}$, any additional terms on the right do not decrease the product by (S3)).

Let $e \in \mathbb{E}$ be arbitrary. We define morphism sets

$$\Psi_A = \{\, \phi \circ \beta' \circ \alpha \circ - = e \,\} \qquad \Psi_B = \{\, \phi \circ \beta' \circ - = e \,\} \qquad \Psi_D = \{\, \phi \circ - = e \,\}$$
$$\Psi_C = \{\, \phi \circ \alpha' \circ - = e \,\}$$

Observe that by $\alpha' \circ \beta = \beta' \circ \alpha$, we have

$$\Psi_A = \{\, \phi \circ \beta' \circ \alpha \circ - = e \,\} = \{\, \phi \circ \alpha' \circ \beta \circ - = e \,\}$$

and so $\beta \circ \Psi_A = \{\, \tau \in \Psi_C \mid \exists \zeta.\ \tau = \beta \circ \zeta \,\}$. Equation (5.1) thus becomes

$$\#\Psi_D = \#\Psi_B + \#\Psi_C - \#(\beta \circ \Psi_A) \tag{5.2}$$

Because $dom(\mathbb{E})$ is traceable along δ and the square δ commutes, we have

$$\Psi_D = \beta' \circ \Psi_B \cup \alpha' \circ \Psi_C \tag{5.3}$$
$$\beta' \circ \Psi_B \cap \alpha' \circ \Psi_C \supseteq \alpha' \circ \beta \circ \Psi_A \tag{5.4}$$

For part (B) of the lemma, we use the general laws

$$\#(A \cup B) \leq \#A + \#B \tag{5.5}$$
$$\#(f \circ A) \leq \#A \tag{5.6}$$
$$(f \circ A - f \circ B) \subseteq f \circ (A - B) \tag{5.7}$$

and proceed as follows:

$$\Psi_D = \beta' \circ \Psi_B \cup (\alpha' \circ \Psi_C - \alpha' \circ \beta \circ \Psi_A) \quad \text{by (5.3) and (5.4)} \tag{5.8}$$
$$\#\Psi_D \leq \#(\beta' \circ \Psi_B) + \#(\alpha' \circ \Psi_C - \alpha' \circ \beta \circ \Psi_A) \quad \text{by (5.8) and (5.5)} \tag{5.9}$$
$$\leq \#\Psi_B + \#(\alpha' \circ \Psi_C - \alpha' \circ \beta \circ \Psi_A) \quad \text{by (5.6)} \tag{5.10}$$
$$\leq \#\Psi_B + \#(\alpha' \circ (\Psi_C - \beta \circ \Psi_A)) \quad \text{by (5.7)} \tag{5.11}$$
$$\leq \#\Psi_B + \#(\Psi_C - \beta \circ \Psi_A) \quad \text{by (5.6)} \tag{5.12}$$
$$= \#\Psi_B + \#\Psi_C - \#(\beta \circ \Psi_A) \quad \text{since } \beta \circ \Psi_A \subseteq \Psi_C \tag{5.13}$$

Equation (5.13) establishes Equation (5.2) for $\leq$, and hence proves part (B).

We now prove part (A) of the lemma. Because $dom(\mathbb{E})$ is vertically strongly traceable along δ, we have $\beta' \circ \Psi_B \cap \alpha' \circ \Psi_C = \alpha' \circ \beta \circ \Psi_A$. Hence from (5.3) we obtain

$$\Psi_D = \beta' \circ \Psi_B \uplus (\alpha' \circ \Psi_C - \alpha' \circ \beta \circ \Psi_A) \tag{5.14}$$
$$\#\Psi_D = \#(\beta' \circ \Psi_B) + \#(\alpha' \circ \Psi_C - \alpha' \circ \beta \circ \Psi_A) \tag{5.15}$$
$$= \#\Psi_B + \#(\alpha' \circ \Psi_C - \alpha' \circ \beta \circ \Psi_A) \tag{5.16}$$

where the last equality is justified by $dom(\mathbb{E})$-monicity of β'.

We have the following property:

($\star$) For any $f, g \in \Psi_C$ with $f \neq g$ and $g \notin \beta \circ \Psi_A$, we have $\alpha' \circ f \neq \alpha' \circ g$.

The validity of this property can be argued as follows:

(i) If $f \notin \beta \circ \Psi_A$, then $\alpha' \circ f \neq \alpha' \circ g$ follows since α' is $dom(\mathbb{E})$-monic outside of β.
(ii) If $f \in \beta \circ \Psi_A$, then there exists $f' \in \Psi_A$ with $f = \beta \circ f'$. For a contradiction, assume that $\alpha' \circ f = \alpha' \circ g$. Then $\alpha' \circ g = \alpha' \circ f = \alpha' \circ \beta \circ f' = \beta' \circ \alpha \circ f'$. From vertical strong traceability, it follows that $g \in \beta \circ \Psi_A$. This contradicts our assumption.

Thus, we have established property ($\star$). From ($\star$) it follows that

$$\#(\alpha' \circ \Psi_C - \alpha' \circ \beta \circ \Psi_A) = \#(\alpha' \circ (\Psi_C - \beta \circ \Psi_A)) \tag{5.17}$$
$$= \#\Psi_C - \#(\beta \circ \Psi_A) \tag{5.18}$$

Then using Equation (5.16), we get

$$\#\Psi_D = \#\Psi_B + \#(\alpha' \circ \Psi_C - \alpha' \circ \beta \circ \Psi_A) \tag{5.19}$$
$$= \#\Psi_B + \#\Psi_C - \#(\beta \circ \Psi_A) \tag{5.20}$$

This establishes Equation (5.2), and hence proves part (A). □

Remark 5.11 (Related work)**.** Lemma 5.10 is closely related to Lemma 2 of [BKNZ15] which establishes $\mathbf{w}_{\mathcal{T}}(\phi) = \mathbf{w}_{\mathcal{T}}(\phi \circ \beta') \odot \mathbf{w}_{\mathcal{T}}(\phi \circ \alpha')$ for directed, edge-labelled multigraphs under the following assumptions:

(i) the interface A must be discrete (contains no edges),
(ii) the weighted type graph has only edge weights (no node weights).

As a consequence of the discrete interface A, all arrows involved in the pushout are edge-monic (Example 5.8).

Lemma 5.10 generalizes Lemma 2 of [BKNZ15] in multiple directions:

(a) it works for arbitrary categories with some (strongly) traceable $dom(\mathbb{E})$,
(b) it does not require discreteness ($dom(\mathbb{E})$-freeness) of the interface, and
(c) it allows for α and α' that are not $dom(\mathbb{E})$-monic.

Both (a) and (b) are crucial to make the termination techique applicable in general categories, and (c) is important to allow for rewrite rules whose right morphism merges elements that we are weighing.

5.5. **Weighing Pushout Objects.** The following lemma (cf. [BKNZ15, Lemma 1]) is a direct consequence of the universal property of pushouts. We will use this property for weighing pushout objects.

Lemma 5.12 (Pushout morphisms)**.** *Let $T \in Ob(\mathbf{C})$ and consider a pushout square*

$$\begin{array}{ccc} A & \xrightarrow{\alpha} & B \\ {\scriptstyle\beta}\downarrow & \delta & \downarrow{\scriptstyle\beta'} \\ C & \xrightarrow{\alpha'} & D \end{array}$$

For every morphism $t_A : A \to T$, there is a bijection θ from

(a) *the class U of morphisms $t_D : D \to T$ for which $t_A = t_D \circ \beta' \circ \alpha$, to*
(b) *the class P of pairs $(t_B : B \to T,\ t_C : C \to T)$ of morphisms for which $t_A = t_B \circ \alpha = t_C \circ \beta$,*

such that for all $t_D \in U$ and pairs $(t_B, t_C) \in P$, we have:

$$\theta(t_D) = (t_B, t_C) \iff t_B = t_D \circ \beta' \text{ and } t_C = t_D \circ \alpha' .$$

Proof. Let $t_A : A \to T$ be given. For $p = (t_B, t_C) \in P$, define $\zeta(p) = t_D$ where $t_D : D \to T$ is the unique morphism satisfying $t_B = t_D \circ \beta'$ and $t_C = t_D \circ \alpha'$. Then $t_A = t_B \circ \alpha = t_D \circ \beta' \circ \alpha$ and so $\zeta(p) \in U$. Map ζ is injective, because if $\zeta(t_B, t_C) = t_D = \zeta(t'_B, t'_C)$, then $t_B = t_D \circ \beta' = t'_B$ and $t_C = t_D \circ \alpha' = t'_C$. It is surjective, because for any $t_D \in U$, $(t_D \circ \beta', t_D \circ \alpha') \in P$ and t_D must be the universal morphism for this pair. Thus ζ and $\theta = \zeta^{-1}$ are bijections. Direction $\implies$ now trivially follows from the commutative property of universal morphisms, and direction $\impliedby$ by using uniqueness of universal morphisms. □

Lemma 5.13 (Weighing pushout objects)**.** *Let $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ be a finitary weighted type graph. Consider an oriented pushout square δ:*

$$\begin{array}{ccc} A & \xrightarrow{\alpha} & B \\ {\scriptstyle\beta}\downarrow & \delta & \downarrow{\scriptstyle\beta'} \\ C & \xrightarrow{\alpha'} & D \end{array}$$

Define

$$k = \bigoplus_{t_A : A \to T} \Big(\bigoplus_{\substack{t_C : C \to T \\ t_A = t_C \circ \beta}} \mathbf{w}_{\mathcal{T}}(t_C - (\beta \circ -)) \Big) \odot \underbrace{\Big(\bigoplus_{\substack{t_B : B \to T \\ t_A = t_B \circ \alpha}} \mathbf{w}_{\mathcal{T}}(t_B) \Big)}_{= \mathbf{w}_{\mathcal{T}}(\{\, - \circ \alpha = t_A \,\})}$$

Then the following holds:

(A) *We have $\mathbf{w}_{\mathcal{T}}(D) = k$ if δ is weighable with $\mathcal{T}$.*
(B) *We have $\mathbf{w}_{\mathcal{T}}(D) \leq k$ if δ is bounded above by $\mathcal{T}$.*

Proof. For part (A) of the lemma we have:

$$
\begin{aligned}
\mathbf{w}_{\mathcal{T}}(D) &= \bigoplus_{t_D:D\to T} \mathbf{w}_{\mathcal{T}}(t_D) \\
&= \bigoplus_{t_A:A\to T} \bigoplus_{\substack{t_D:D\to T \\ t_A=t_D\circ\beta'\circ\alpha}} \mathbf{w}_{\mathcal{T}}(t_D) \\
&= \bigoplus_{t_A:A\to T} \bigoplus_{\substack{t_D:D\to T \\ t_A=t_D\circ\beta'\circ\alpha}} \mathbf{w}_{\mathcal{T}}(t_D\circ\beta') \odot \mathbf{w}_{\mathcal{T}}(t_D\circ\alpha' - (\beta\circ -)) && \text{by Lemma 5.10} \\
&= \bigoplus_{t_A:A\to T} \bigoplus_{\substack{t_C:C\to T \\ t_A=t_C\circ\beta}} \bigoplus_{\substack{t_B:B\to T \\ t_A=t_B\circ\alpha}} \mathbf{w}_{\mathcal{T}}(t_B) \odot \mathbf{w}_{\mathcal{T}}(t_C - (\beta\circ -)) && \text{by Lemma 5.12} \\
&\overset{(\dagger)}{=} \bigoplus_{t_A:A\to T} \bigoplus_{\substack{t_C:C\to T \\ t_A=t_C\circ\beta}} \Big(\mathbf{w}_{\mathcal{T}}(t_C - (\beta\circ -)) \odot \bigoplus_{\substack{t_B:B\to T \\ t_A=t_B\circ\alpha}} \mathbf{w}_{\mathcal{T}}(t_B)\Big) \\
&\overset{(\dagger)}{=} \bigoplus_{t_A:A\to T} \Big(\bigoplus_{\substack{t_C:C\to T \\ t_A=t_C\circ\beta}} \mathbf{w}_{\mathcal{T}}(t_C - (\beta\circ -))\Big) \odot \Big(\bigoplus_{\substack{t_B:B\to T \\ t_A=t_B\circ\alpha}} \mathbf{w}_{\mathcal{T}}(t_B)\Big)
\end{aligned}
$$

The equalities marked with (†) are justified since $\mathbf{w}_{\mathcal{T}}(t_C - (\beta\circ -))$ and $\bigoplus_{\substack{t_B:B\to T \\ t_A=t_B\circ\alpha}} \mathbf{w}_{\mathcal{T}}(t_B)$ are constant.

For part (B) we get a similar derivation. The only difference concerns the step with the application of Lemma 5.10 which yields $\leq$ instead of $=$ under the assumptions for (B). □

6. Termination via Weighted Type Graphs

In the previous section, we have developed techniques for weighing pushout objects. Our next goal is to apply these techniques to the pushout squares of double pushout diagrams in order to prove that rewrite steps reduce the weight.

As we have already noted in Remark 3.2, there exist many variations of the general concept of double pushout graph transformation. To keep our treatment as general as possible, we introduce in Section 6.1 an abstract concept of a double pushout framework $\mathfrak{F}$ which maps rules $\rho : L \leftarrow K \rightarrow R$ to classes $\mathfrak{F}(\rho)$ of double pushout diagrams with top span ρ.

In order to prove that rewrite steps decrease the weight, we need to ensure that every object that can be rewritten admits some morphism into the weighted type graph. To this end, we introduce the concept of context closures of rules in Section 6.2.

In Section 6.3, we prove that rewrite steps are decreasing (Theorem 6.11) if the rules satisfy certain criteria (Definition 6.9). Finally, we state our main theorem (Theorem 6.12) for proving (relative) termination.

6.1. **Double Pushout Frameworks.**

Definition 6.1. A **double pushout diagram** δ is a diagram of the form

$$\begin{array}{ccccc} L & \leftarrow l - & K & -r \rightarrow & R \\ \downarrow m & \text{PO} & \downarrow u & \text{PO} & \downarrow w \\ G & \leftarrow l' - & C & -r' \rightarrow & H \end{array}$$

This diagram δ is a **witness** for the **rewrite step** $G \Rightarrow^{\delta} H$. We use the following notation for the left and the right square of the diagram, respectively:

$$\mathit{left}(\delta) = \langle\, C \overset{u}{\leftarrow} K \overset{l}{\rightarrow} L;\ C \overset{l'}{\rightarrow} G \overset{m}{\leftarrow} L \,\rangle$$

$$\mathit{right}(\delta) = \langle\, C \overset{u}{\leftarrow} K \overset{r}{\rightarrow} R;\ C \overset{r'}{\rightarrow} H \overset{w}{\leftarrow} R \,\rangle$$

Both $\mathit{left}(\delta)$ and $\mathit{right}(\delta)$ are oriented pushout squares (where u is vertical).

For a class Δ of double pushout diagrams, we define $\mathit{left}(\Delta) = \{\, \mathit{left}(\delta) \mid \delta \in \Delta \,\}$ and $\mathit{right}(\Delta) = \{\, \mathit{right}(\delta) \mid \delta \in \Delta \,\}$.

Definition 6.2 (Double pushout framework)**.** A **double pushout framework** $\mathfrak{F}$ is a mapping of DPO rules to classes of DPO diagrams such that, for every DPO rule ρ, $\mathfrak{F}(\rho)$ is a class of DPO diagrams with top-span ρ.

The rewrite relation $\Rightarrow_{\rho,\mathfrak{F}}$ induced by a DPO rule ρ in $\mathfrak{F}$ is defined as follows: $G \Rightarrow_{\rho,\mathfrak{F}} H$ iff $G \Rightarrow^{\delta} H$ for some $\delta \in \mathfrak{F}(\rho)$. The rewrite relation $\Rightarrow_{\mathfrak{S},\mathfrak{F}}$ induced by a set $\mathfrak{S}$ of DPO rules is given by: $G \Rightarrow_{\mathfrak{S},\mathfrak{F}} H$ iff $G \Rightarrow_{\rho,\mathfrak{F}} H$ for some $\rho \in \mathfrak{S}$. When $\mathfrak{F}$ is clear from the context, we suppress $\mathfrak{F}$ and write $\Rightarrow_{\rho}$ and $\Rightarrow_{\mathfrak{S}}$.

6.2. **Context Closures.** For proving termination, we need to establish that every rewrite step causes a decrease in the weight of the host graph. In particular, we need to ensure that every host graph admits some morphism into the weighted type graph. For this purpose, we introduce the concept of a context closure. Intuitively, a context closure for a rule $L \leftarrow K \rightarrow R$ is a morphism $c : L \rightarrow T$ into the type graph T in such a way that every match $L \rightarrow G$ can be mapped 'around c'.

Definition 6.3 (Context closure)**.** Let $\mathcal{A}$ be a class of morphisms. A **context closure** of an object $L \in \mathit{Ob}(\mathbf{C})$ for class $\mathcal{A}$ is an arrow $c : L \rightarrow T$ such that, for every arrow $\alpha : L \rightarrow G$ in $\mathcal{A}$, there exists a morphism $\beta : G \rightarrow T$ with $c = \beta \circ \alpha$. This can be depicted as

$$\begin{array}{ccc} L & \xrightarrow{\quad\alpha\quad} & G \\ & {\scriptstyle c}\searrow \quad = \quad \swarrow{\scriptstyle \beta} & \\ & T & \end{array}$$

Remark 6.4 (Partial map classifiers as context closures)**.** The notion of a context closure is strictly weaker than a partial map classifier. If category $\mathbf{C}$ has an $\mathcal{M}$-partial map classifier (T, η) (see [AHS06, Definition 28.1] and [CDE+20, Definition 5]) for a stable class of monics $\mathcal{M}$, then the $\mathcal{M}$-partial map classifier arrow $\eta_X : X \hookrightarrow T(X)$ for an object X is a context closure of X for $\mathcal{M}$, because for any $\alpha : X \hookrightarrow Z$ in $\mathcal{M}$ there exists a unique $\beta : Z \rightarrow T(X)$ making the following square a pullback:

$$\begin{array}{ccc} X & = & X \\ {\scriptstyle \eta_X}\downarrow & \text{PB} & \downarrow{\scriptstyle \alpha} \\ T(X) & \leftarrow \beta & Z \end{array}$$

Definition 6.5 (Context closure for a rule)**.** Let $\mathfrak{F}$ be a DPO framework and ρ a DPO rule $L \leftarrow K \rightarrow R$. A **context closure for ρ and T (in $\mathfrak{F}$)** is a context closure $c : L \to T$ for the class of match morphisms occurring in $\mathfrak{F}(\rho)$.

Remark 6.6 (Context closures for unrestricted matching)**.** Consider a DPO rule $\rho = L \leftarrow K \rightarrow R$ in an arbitrary DPO framework $\mathfrak{F}$ (in particular, matching can be unrestricted). Every morphism $c : L \to 1 \to T$, that factors through the terminal object 1, is a context closure for ρ and T in $\mathfrak{F}$.

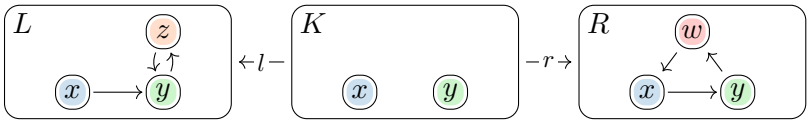
This particular choice of a context closure generalizes the flower nodes from [BKNZ15] which are defined specifically for the category of edge-labelled multigraphs. A flower node [BKNZ15] in a graph T is just a morphism $1 \to T$.

When employing such context closures, we establish termination with respect to unrestricted matching. This is bound to fail if termination depends on the matching restrictions.

Remark 6.7 (Increasing the power for monic matching)**.** For rules $L \overset{l}{\leftarrow} K \overset{r}{\rightarrow} R$ with monic matching, it is typically fruitful to choose a type graph T and context closure $c : L \to T$ in such a way that c avoids collapsing $l(K)$ in L. In other words, the context closure c should be monic for $\{\, l \circ h \mid h \in \mathrm{Hom}(-, K) \,\}$. In the case that l is monic, this is equivalent to $c \circ l$ being monic.

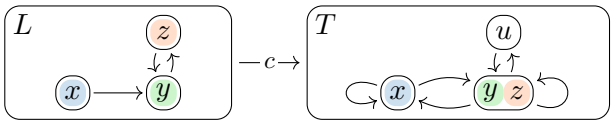
For **Graph**, the intuition is as follows. When collapsing nodes of $l(K)$ along c, we actually prove a stronger termination property than intended. We then prove termination even if matches $m : L \to G$ are allowed to collapse those nodes.

Example 6.8 (Context closure for monic matching)**.** Let ρ be the DPO rule



from [OE24b, Example 6.1] in **Graph** with a framework $\mathfrak{F}$ with monic matching.

Following Remark 6.7, we construct a context closure $c : L \to T$ for ρ that avoids collapsing $l(K)$ of L. Since $l(K) \cong K$, this means that the complete graph around the nodes of K must be a subobject of T. We pick



Indeed, for every monic match $m : L \rightarrowtail G$ there exists a morphism $\alpha : G \to T$ such that $c = \alpha \circ m$. Thus c is a context closure for ρ and T in $\mathfrak{F}$.

6.3. **Proving Termination.** In this section we prove our main theorem (Theorem 6.12) for using weighted type graphs to prove termination of graph transformation systems.

First, we introduce different notions of decreasing rules (Defintion 6.9). Then we prove that rewrite steps arising from these rules cause a decrease in the weight of the host graphs (Theorem 6.11). Finally, we summarize our technique for proving (relative) termination (Theorem 6.12).

The following definition introduces the notions of weakly, uniformly and closure decreasing rules. Weakly decreasing rules never increase the weight of the host graph. Uniformly decreasing rules reduce the weight of the host graph for every mapping of the host graph onto

the type graph. If the underlying semiring is strictly monotonic, then closure decreasingness suffices: the weight of the host graph decreases for those mappings that correspond to the context closure of the rule (and is non-increasing for the other mappings).

Definition 6.9 (Decreasing rules)**.** Let $\mathfrak{F}$ be a DPO framework. Moreover, let $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ be a finitary weighted type graph over a well-founded commutative semiring $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1}, \prec, \leq \rangle$. A DPO rule $\rho : L \overset{l}{\leftarrow} K \overset{r}{\rightarrow} R$ is called

(i) **weakly decreasing** (w.r.t. $\mathcal{T}$ in $\mathfrak{F}$) if
- $\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \geq \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$ for every $t_K : K \to T$;

(ii) **uniformly decreasing** (w.r.t. $\mathcal{T}$ in $\mathfrak{F}$) if
- for every $t_K : K \to T$:
  $\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \succ \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$ or
  $\{ - \circ l = t_K \} = \varnothing = \{ - \circ r = t_K \}$, and
- there exists a context closure $\mathfrak{c}_\rho$ for ρ and T in $\mathfrak{F}$;

If semiring S is moreover strictly monotonic, we say that ρ is

(iii) **closure decreasing** (w.r.t. $\mathcal{T}$ in $\mathfrak{F}$) if
- ρ is weakly decreasing,
- there exists a context closure $\mathfrak{c}_\rho$ for ρ and T in $\mathfrak{F}$, and
- $\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \succ \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$ for $t_K = \mathfrak{c}_\rho \circ l$.

By saying that a rule is closure decreasing, we tacitly assume that the semiring is strictly monotonic (for otherwise the concept is not defined). Note that if S is strictly monotonic, uniformly decreasing implies closure decreasing.

Remark 6.10. Our notions of uniformly and closure decreasing rules generalize the corresponding concepts (strongly and strictly decreasing rules) in [BKZ14, BKNZ15].

In [BKNZ15], the notion of 'strongly decreasing' rules requires that

$$\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \succ \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$$

for every $t_K : K \to T$. Bruggink et al. [BKNZ15] claim that their notion generalizes the termination techniques from their earlier work [BKZ14] focused on the tropical and arctic semirings. However, this is not the case since there typically exist morphisms $t_K : K \to T$ for which $\{ - \circ l = t_K \} = \varnothing$ and thus $\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) = \mathbf{0}$:

(1) In the arctic semiring $\mathbf{0} = -\infty$ and $-\infty \succ \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$ is impossible. In contrast, the paper [BKZ14] allows for $\mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \}) = -\infty$ in this case.
(2) In the tropical semiring $\mathbf{0} = \infty$ and $\infty \succ \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$ enforces that $\{ - \circ r = t_K \} \neq \varnothing$ while [BKZ14] allows for $\mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \}) = \infty$ in this case.

Our notion of uniform decreasingness properly generalizes the techniques in [BKZ14] and works for any well-founded commutative semiring.

Theorem 6.11 (Decreasing steps)**.** *Let $\mathfrak{F}$ be a DPO framework. Moreover, let $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ be a finitary weighted type graph over a well-founded commutative semiring $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1}, \prec, \leq \rangle$. Let ρ be a DPO rule and $\delta \in \mathfrak{F}(\rho)$ such that*
- *left(δ) is weighable with $\mathcal{T}$, and*
- *right(δ) is bounded above by $\mathcal{T}$.*

Then, for the induced rewrite step $G \Rightarrow^{\delta} H$, we have
(1) *$\mathbf{w}_{\mathcal{T}}(G) \geq \mathbf{w}_{\mathcal{T}}(H)$ if ρ is weakly decreasing;*
(2) *$\mathbf{w}_{\mathcal{T}}(G) \succ \mathbf{w}_{\mathcal{T}}(H)$ if ρ is uniformly or closure decreasing.*

Proof of Theorem 6.11. Let δ be the following double pushout diagram:

$$\begin{array}{ccccc} L & \xleftarrow{l} & K & \xrightarrow{r} & R \\ \downarrow{\scriptstyle m} & \text{PO} & \downarrow{\scriptstyle u} & \text{PO} & \downarrow{\scriptstyle w} \\ G & \xleftarrow{l'} & C & \xrightarrow{r'} & H \end{array}$$

For $t_K : K \to T$, define

$$w_{C,t_K} = \bigoplus_{\substack{t_C : C \to T \\ t_K = t_C \circ u}} \mathbf{w}_{\mathcal{T}}(t_C - (u \circ -))$$

By Lemma 4.5, $\mathbf{1} \leq \mathbf{w}_{\mathcal{T}}(t_C - (u \circ -)) \neq \mathbf{0}$. By Proposition 3.7, for all $x, y \in S$:

(a) $x \leq y \implies w_{C,t_K} \odot x \leq w_{C,t_K} \odot y$;
(b) $x \prec y \implies w_{C,t_K} \odot x \prec w_{C,t_K} \odot y$ if there exists t_C with $t_K = t_C \circ u$;
(c) $w_{C,t_K} \odot x = \mathbf{0} = w_{C,t_K} \odot y$ if there does not exist t_C with $t_K = t_C \circ u$;
(d) If there is a context closure $\mathfrak{c}_\rho$ for ρ and T in $\mathfrak{F}$, then

$$x \prec y \implies w_{C,t_K} \odot x \prec w_{C,t_K} \odot y$$

for $t_K = \mathfrak{c}_\rho \circ l$. This can be argued as follows. By the definition of context closure, there exists $\alpha : G \to T$ such that $\mathfrak{c}_\rho = \alpha \circ m$. Define $t_C = \alpha \circ l'$. Then $t_K = \alpha \circ m \circ l = \alpha \circ l' \circ u = t_C \circ u$ and the claim follows from (b).

Since *left*(δ) is weighable with $\mathcal{T}$, by Lemma 5.13 we obtain

$$\mathbf{w}_{\mathcal{T}}(G) = \bigoplus_{t_K : K \to T} w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \})$$

Since *right*(δ) is bounded above by $\mathcal{T}$, by Lemma 5.13 we obtain

$$\mathbf{w}_{\mathcal{T}}(H) \leq \bigoplus_{t_K : K \to T} w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$$

We distinguish cases:

(1) If ρ is weakly decreasing, then $\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \geq \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$ for every $t_K : K \to T$. By (a) we have:

$$w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \geq w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \}) \tag{6.1}$$

for every $t_K : K \to T$. Hence $\mathbf{w}_{\mathcal{T}}(G) \geq \mathbf{w}_{\mathcal{T}}(H)$ by (S1).

(2) If ρ is uniformly decreasing, then from (b) and (c) we obtain
   (i) $w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \succ w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$, or
   (ii) $w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) = \mathbf{0} = w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$.

for every $t_K : K \to T$. To establish $\mathbf{w}_{\mathcal{T}}(G) \succ \mathbf{w}_{\mathcal{T}}(H)$, using (S2), it suffices to show that we have case (i) for some $t_K : K \to T$. This follows from (d) since we have a context closure $\mathfrak{c}_\rho$ for ρ and T by assumption.

(3) If ρ is closure decreasing, then it is also weakly decreasing and we obtain (6.1) for every $t_K : K \to T$. Since the semiring is strictly monotonic, by (S5), it suffices to show that there exists some $t_K : K \to T$ such that

$$w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \succ w_{C,t_K} \odot \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \}) \tag{6.2}$$

in order to conclude $\mathbf{w}_{\mathcal{T}}(G) \succ \mathbf{w}_{\mathcal{T}}(H)$. There is a context closure $\mathfrak{c}_\rho$ for ρ and T. By assumption $\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \succ \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$ for $t_K = \mathfrak{c}_\rho \circ l$, and we obtain (6.2) by (d). □

We now state our main theorem of proving (relative) termination using weighted type graphs.

Theorem 6.12 (Proving termination)**.** *Let $\mathfrak{F}$ be a DPO framework and let $\mathfrak{S}_1$ and $\mathfrak{S}_2$ be sets of double pushout rules. Let $\langle S, \oplus, \odot, \mathbf{0}, \mathbf{1}, \prec, \leq \rangle$ be a well-founded commutative semiring, and let $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ be a finitary weighted type graph such that*

- *left(δ) is weighable with $\mathcal{T}$, and*
- *right(δ) is bounded above by $\mathcal{T}$*

for every rule $\rho \in (\mathfrak{S}_1 \cup \mathfrak{S}_2)$ and double pushout diagram $\delta \in \mathfrak{F}(\rho)$. If

(1) *ρ is weakly decreasing for every $\rho \in \mathfrak{S}_2$, and*
(2) *ρ is uniformly or closure decreasing for every $\rho \in \mathfrak{S}_1$,*

then $\mathfrak{S}_1$ is terminating relative to $\mathfrak{S}_2$.

Proof of Theorem 6.12. Follows immediately from Theorem 6.11. □

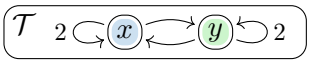
7. Examples

Notation 7.1 (Visual notation)**.** *We use the visual notation as in* [OE24b]. *The vertices of graphs are non-empty sets $\{x_1, \ldots, x_n\}$ depicted by boxes $x_1 \cdots x_n$. We will choose the vertices of the graphs in such a way that the homomorphisms $h : G \to H$ are fully determined by $S \subseteq h(S)$ for all $S \in V_G$. For instance, in Example 7.3 below, the morphism $\mathfrak{c}_\rho : L \to T$ maps nodes $\{x\}, \{y\}, \{z\} \in L$ as follows: $\mathfrak{c}_\rho(\{x\}) = \{x\}$ and $\mathfrak{c}_\rho(\{y\}) = \mathfrak{c}_\rho(\{z\}) = \{y, z\} \in T$.*

We start with two examples (Examples 7.2 and 7.3) for which the approaches from [BKZ14, BKNZ15] fail, as the systems are not terminating with unrestricted matching.

Example 7.2 (Loop unfolding with monic matching)**.** Let ρ be the DPO rule

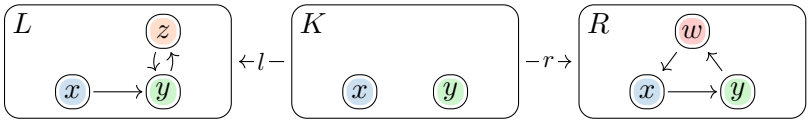
in **Graph**, and let matches m restrict to monic matches. We prove termination using the weighted type graph



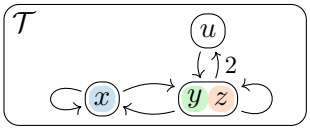
over the (strictly monotonic) arithmetic semiring. The type graph T is unlabelled. The numbers along the edges indicate the weighted elements $\mathbb{E}$. Here $\mathbb{E}$ consists of 2 morphisms with domain $\bullet \longrightarrow \bullet$ (see further Remark 5.4) and their weight is the number along the edge they target. So $\mathbb{E} = \{ e_x, e_y \}$ where e_x and e_y map $\bullet \longrightarrow \bullet$ onto the loop on $\{x\}$ and the loop on $\{y\}$ in T, respectively. The weights of e_x and e_y are $\mathbf{w}(e_x) = \mathbf{w}(e_y) = 2$.

As the context closure $\mathfrak{c}_\rho$ for ρ and T we use the inclusion $\mathfrak{c}_\rho : L \rightarrowtail T$. Clearly, for any monic match $L \rightarrowtail G$, there exists a map $\alpha : G \to T$ such that $\mathfrak{c}_\rho = \alpha \circ m$. For $t_K = \mathfrak{c}_\rho \circ l$ we have $\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) = 2 \succ 1 = \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$ since the empty product is 1. Moreover, $\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) \geq \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$ for all other $t_K : K \to T$. Thus ρ is closure decreasing, and consequently terminating by Theorem 6.12.

Example 7.3 (Reconfiguration)**.** Continuing Example 6.8, let ρ be the DPO rule



from [OE24b, Example 6.1] in **Graph**, and let matches m restrict to monic matches. We prove termination of ρ with matches m restrict to monic matches. We use the weighted type graph

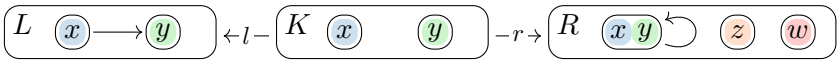


over the (strictly monotonic) arithmetic semiring. This is the type graph constructed in Example 6.8, enriched with weighted elements. $\mathbb{E}$ consists of a single morphism with weight 2, mapping $\bullet \longrightarrow \bullet$ onto the edge from $\{y, z\}$ to $\{u\}$ in T.

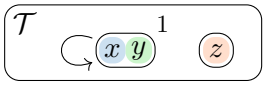
As the context closure $\mathfrak{c}_\rho$ for ρ and T we use $\mathfrak{c}_\rho : L \to T$ as indicated by the node names ($\{y\}$ and $\{z\}$ are mapped onto $\{y, z\}$). For $t_K = \mathfrak{c}_\rho \circ l$ we then obtain $\mathbf{w}_\mathcal{T}(\{ - \circ l = t_K \}) = 1 + 1 + 2 \succ 1 + 1 = \mathbf{w}_\mathcal{T}(\{ - \circ r = t_K \})$ since the empty product is 1, and $\{z\}$ in L can be mapped onto $\{x\}$, $\{y, z\}$ and $\{u\}$, while $\{w\}$ in R can only be mapped onto $\{x\}$ and $\{y, z\}$ in T. Furthermore, $\mathbf{w}_\mathcal{T}(\{ - \circ l = t_K \}) \geq \mathbf{w}_\mathcal{T}(\{ - \circ r = t_K \})$ for all other $t_K : K \to T$. Thus ρ is closure decreasing, and hence terminating by Theorem 6.12.

We continue with an example (Example 7.4) of rewriting simple graphs. Here the techniques from [BKZ14, BKNZ15] are not applicable, because they are defined only for multigraphs. Even when considering the rule as a rewrite rule on multigraphs, the technique from [BKZ14] is not applicable due to non-monic r, and the technique from [BKNZ15] fails because the rule is not terminating with respect to unrestricted matching.

Example 7.4 (Simple graphs and monic matching)**.** Let ρ be the DPO rule



in the category of simple graphs **SGraph**, and let matches m restrict to monic matches. This rule folds an edge into a loop, similar to [OE24b, Example 5.2], but extended with the addition of two fresh nodes in the right-hand side. We prove termination using the weighted type graph $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ given by



over the tropical semiring. So $\mathbb{E}$ consists of a single morphism e mapping $\bullet$ onto x in T, and $\mathbf{w}(e) = 1$. (See Remark 5.5 on strong traceability in **SGraph**.) As the context closure $\mathfrak{c}_\rho$ for ρ and T we use the morphism $\mathfrak{c}_\rho : L \to T$ as indicated by the node names ($\{x\}$ and $\{y\}$ are mapped onto $\{x, y\}$).

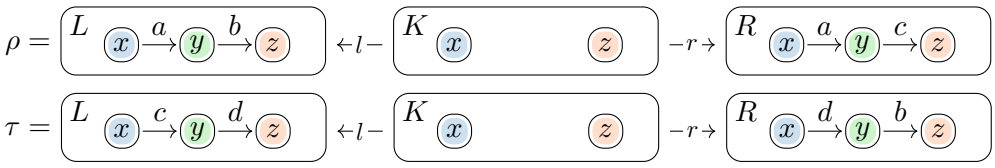
Then $\mathbf{w}_\mathcal{T}(\{ - \circ l = t_K \}) = 2 \succ 1 = 3 \oplus 2 \oplus 2 \oplus 1 = \mathbf{w}_\mathcal{T}(\{ - \circ r = t_K \})$ for $t_K = \mathfrak{c}_\rho \circ l$. For every other morphism $t_K : K \to T$ it holds that $\{ - \circ l = t_K \} = \varnothing = \{ - \circ r = t_K \}$. Thus ρ is uniformly decreasing, and Theorem 6.12 yields termination of ρ.

Example 7.5 (Reconfiguration on simple graphs)**.** Example 7.3 can also be considered on **SGraph** with monic matching. Assume that we add in K an edge from $\{x\}$ to $\{y\}$ (this does not change the rewrite relation). Then l is regular monic, and we have strong traceability for the edges along the left pushout squares of rewrite steps. Hence all left squares are

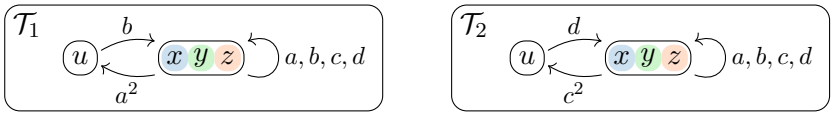
weighable, and all right squares are bounded above by the type graph in Example 7.3; so the same type graph proves also termination on **SGraph**.

We now consider some examples in **Graph** using unrestricted matching.

Example 7.6 (Unrestricted matching)**.** Consider the following double pushout rewrite system



from [Plu18, Example 3] and [Plu95, Example 3.8] in **Graph** with unrestricted matching. These are graph transformation versions of the string rewrite rules $ab \to ac$ and $cd \to db$, respectively. We prove termination in two steps using the type graphs



over the arithmetic semiring. Here the weights of the weighted elements are indicated as superscripts (both type graphs have one weighted element with domain $\bullet \longrightarrow \bullet$ and weight 2).

First, we take the weighted type graph $\mathcal{T}_1$. Consider the rule ρ together with the context closure $\mathfrak{c}_\rho : L \to T_1$ mapping all nodes onto $\{x, y, z\}$ in T_1. For $t_K = \mathfrak{c}_\rho \circ l$ we have

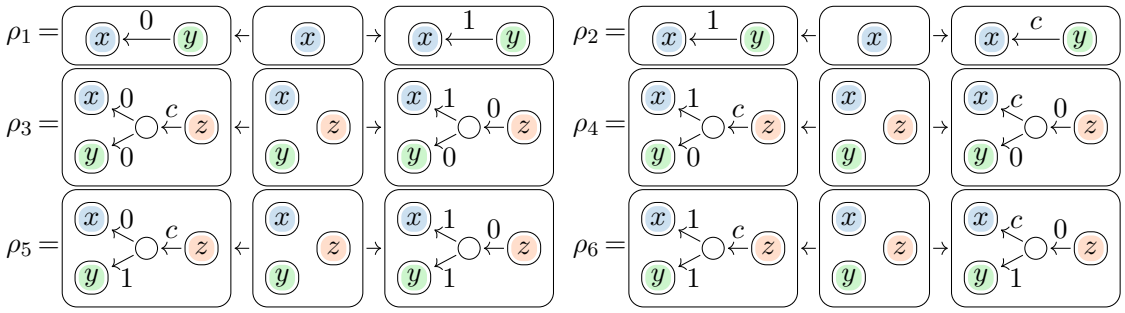
$$\mathbf{w}_{\mathcal{T}_1}(\{ - \circ l = t_K \}) = 1 + 2 \succ 1 = \mathbf{w}_{\mathcal{T}_1}(\{ - \circ r = t_K \}).$$

Here weight 1 arises from those morphisms mapping all nodes onto $\{x, y, z\}$ in T_1; then there are no weighted elements in the image of the morphism and the empty product is 1. For all other $t_K : K \to T$ we have $\mathbf{w}_{\mathcal{T}_1}(\{ - \circ l = t_K \}) = 0 \geq 0 = \mathbf{w}_{\mathcal{T}_1}(\{ - \circ r = t_K \})$. Thus ρ is closure decreasing.

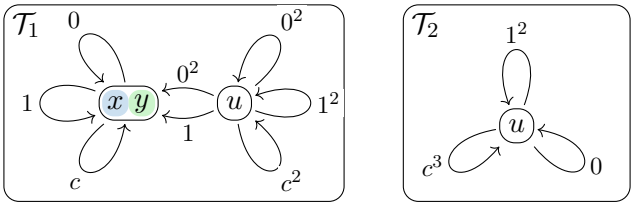
For the rule τ, L and R admit only one morphism with codomain T_1. So it is easy to see that $\mathbf{w}_{\mathcal{T}_1}(\{ - \circ l = t_K \}) \geq \mathbf{w}_{\mathcal{T}_1}(\{ - \circ r = t_K \})$ for all $t_K : K \to T$, and thus τ is weakly decreasing.

As a consequence, by Theorem 6.12, ρ is terminating relative to τ. Hence it suffices to prove termination of τ which follows analogously using $\mathcal{T}_2$.

Example 7.7 (Tree counter)**.** Consider the system



from [BKNZ15, Example 6] in **Graph** with unrestricted matching. This system implements a counter on trees. We prove termination in two steps using the type graphs



over the arithmetic semiring. The superscripts indicate the weights of the weighted elements (all weighted elements have domain $\bullet \longrightarrow \bullet$).

First, we take the weighted type graph $\mathcal{T}_1$. Consider the rule ρ_1 together with the context closure $\mathfrak{c}_\rho : L \to T_1$ mapping all nodes onto $\{x, y\}$ in T_1. For $t_K = \mathfrak{c}_\rho \circ l$ we have

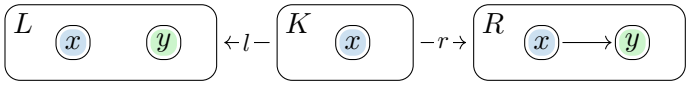
$$\mathbf{w}_{\mathcal{T}_1}(\{ - \circ l = t_K \}) = 1 + 2 \succ 1 + 1 = \mathbf{w}_{\mathcal{T}_1}(\{ - \circ r = t_K \}).$$

For t_K mapping $\{x\}$ to $\{u\}$, we get $\mathbf{w}_{\mathcal{T}_1}(\{ - \circ l = t_K \}) = 2 \geq 2 = \mathbf{w}_{\mathcal{T}_1}(\{ - \circ r = t_K \})$. Thus ρ_1 is closure decreasing. Likewise, ρ_2 is also closure decreasing and all other rules are weakly decreasing. By Theorem 6.12 it suffices to prove termination without the rules ρ_1 and ρ_2.

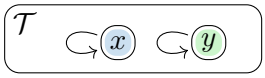
Termination of the remaining rules is established by the weighted type graph $\mathcal{T}_2$. The rules ρ_3, ρ_4, ρ_5 and ρ_6 are all closure decreasing with respect to $\mathcal{T}_2$. As a consequence, the system is terminating by Theorem 6.12.

The following example shows that the technique also can be fruitfully applied for rules $\rho : L \leftarrow K \to R$ for which L is a subgraph of R, that is, $L \rightarrowtail R$.

Example 7.8 (Morphism counting)**.** Let ρ be the double pushout rule



in the category of graphs **Graph** with unrestricted matching. Intuitively, this rule connects an isolated node with an edge. We prove termination using the weighted type graph $\mathcal{T} = (T, \mathbb{E}, S, \mathbf{w})$ given by



over the arithmetic semiring. Here $\mathbb{E} = \varnothing$. This means that every morphism into T has weight 1 (the empty product) and thus the weight of an object $X \in Ob(\mathbf{C})$ is $\#\mathrm{Hom}(X, T)$, the number of morphisms from X to T. As the context closure $\mathfrak{c}_\rho$ for ρ and T we use the inclusion $\mathfrak{c}_\rho : L \rightarrowtail T$ as indicated by the node names.

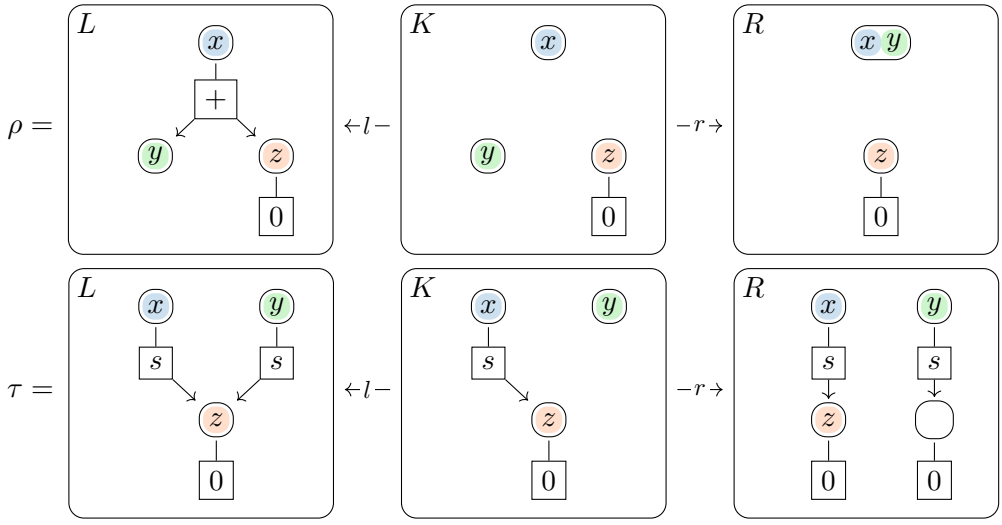
For every morphism $t_K : K \to T$ we have

$$\mathbf{w}_{\mathcal{T}}(\{ - \circ l = t_K \}) = 1 + 1 = 2 \succ 1 = \mathbf{w}_{\mathcal{T}}(\{ - \circ r = t_K \})$$

Thus ρ is closure decreasing, and Theorem 6.12 yields termination of ρ.

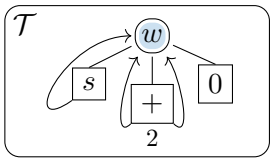
Finally, we consider a hypergraph example with unrestricted matching. Our method can prove relative termination of the first rule, but termination of the second rule does not seem to be provable with our method.

Example 7.9 (Limitations)**.** Consider the following hypergraph rewriting system



from [Plu18, Example 6] with unrestricted matching.

Termination of ρ relative to τ follows using the following weighted type graph



over the arithmetic semiring by Theorem 6.12. Here the '+' hyperedge has weight 2.

However, τ is an interesting rule for which we could not find a termination proof using weighted type graphs over the arctic, tropical or arithmetic semiring. First, note that there exists an epimorphism $e : R \twoheadrightarrow L$ with $l = e \circ r$. In such situations, no weighted type graph over the arithmetic semiring can make the rule strictly decreasing. Second, it seems that the arctic and tropical semiring are also not applicable here. Intuitively, one might try to use the tropical semiring since the additional freedom in mapping R could lead to a lower weight. However, for this rule, there are morphisms $h_1, h_2 : L \to R$ where h_1 maps L onto the left component of R, and h_2 maps L onto the right component of R. So, for any mapping $t : R \to T$, either $t \circ h_1$ or $t \circ h_2$ will be at least as 'light'.

In [Plu18], it is argued that this rule is terminating since the sum of the squares of in-degrees is decreasing in each step. This argument can be automated, for instance, using the technique proposed in [OE24b, Example 5.9]: see Section 9 for a related discussion.

Example 7.9 illustrates a limitation of our technique:

Open Question 7.10. *We conjecture that our technique cannot be applied for rules $\rho : L \overset{l}{\leftarrow} K \overset{r}{\rightarrow} R$ for which there exists an epimorphism $e : R \twoheadrightarrow L$ with $l = e \circ r$. Can the technique be enhanced to deal with such rules?*

8. Implementation and Benchmarks

We have implemented our techniques in graphTT-wtg (short for **Graph T**ermination **T**ool – **w**eighted **t**ype **g**raphs) for automatically proving termination of graph transformation.

The tool is written in Scala and employs Z3 [dMB08] to solve the constraint systems.[3] The source code (including examples) are available for download at

`http://joerg.endrullis.de/downloads/graphTT_wtg.tar.gz`

The tool allows to specify the underlying category using **copresheaves**, i.e., functors $F : \mathbf{I} \to \mathbf{Set}$[4], which are equivalent to **unary algebras** [Löw93] and $\mathcal{C}$-sets [BPHF22]. The grammar for specifying the index category $\mathbf{I}$ is

$$\begin{aligned} \mathit{IndexCategory} &= \mathit{Object}\ \ldots\ \mathit{Object} \\ \mathit{Object} &= \mathit{Name}[[\underbrace{\mathit{Label}, \ldots, \mathit{Label}}_{\text{labels}}]](\underbrace{\mathit{Name}, \ldots, \mathit{Name}}_{\text{arguments}})[!] \end{aligned}$$

where *Name* is the name of an object in $\mathbf{I}$, and the 'arguments' are the targets of outgoing arrows. Optionally, an object can be equipped with a set of labels which corresponds to a slice category construction. graphTT-wtg can thereby be applied to the multigraph examples [BKZ14, BKNZ15], and the hypergraph examples [Plu18].

Moreover, the tool supports what could be called 'partially simple' presheaves in which some of the element sets are simple (the elements are uniquely determined by their 'argument values'). To this end, an object declaration can be followed by '!'. For instance

$$V \qquad \mathrm{edge}(V, V) \qquad \mathrm{plus}(V, V, V)! \qquad \mathrm{flag}[a, b](V)$$

describes presheaves having a set of nodes V, binary edges, ternary hyperedges 'plus' between nodes of V, and unary hyperedges 'flag' labelled with labels from $\{a, b\}$. The binary edges allow for multiple parallel edges. The ternary 'plus' is declared as simple, meaning that there cannot be two 'plus' edges between the same tuple of nodes (the order matters).

graphTT-wtg supports a simple strategy language of the form

Strategy =	
Strategy **;** *Strategy*	sequential composition
| *Strategy* | *Strategy*	parallel composition
| repeat(*Strategy*)	repeat while successful
| arctic(size = $\mathbb{N}$, bits = $\mathbb{N}$, timeout = $\mathbb{N}$)	arctic type graphs
| tropical(size = $\mathbb{N}$, bits = $\mathbb{N}$, timeout = $\mathbb{N}$)	tropical type graphs
| arithmetic(size = $\mathbb{N}$, bits = $\mathbb{N}$, timeout = $\mathbb{N}$)	arithmetic type graphs

This enables the user to specify what technique is applied in what order and for how long. Each strategy can transform the system by removing rules. A strategy is considered successful if at least one rule has been removed. For the basis strategies, that is, arctic, tropical or arithmetic weighted type graphs, 'size' specifies the size of the type graph, 'bits' the number of bits for encoding the semiring elements, and 'timeout' is the the maximum number of seconds to try this technique.

Table 1 shows benchmarks for running graphTT-wtg on the examples from [BKZ14, BKNZ15, Plu95, Plu18][5] under different combinations of the arctic, tropical and arithmetic

[3] graphTT-wtg is a fork of an early version of graphTT [OE24a] described in [OE24b, Section 7]. The two have since diverged.

[4] We restrict to acyclic copresheaves, i.e., copresheaves that are acyclic if one ignores the identity morphisms.

[5] We have included all uniform termination examples from [BKZ14, BKNZ15, Plu95, Plu18]. We did not include [BKZ14, Example 3] as this concerns local termination on a particular starting language. We have

	A	T	N	A,T	A,N	T,N	**A,T,N**
Example 7.2			0.08		0.06	0.06	**0.09**
Example 7.3			0.12		0.11	0.11	**0.12**
Example 7.4		0.03		0.03		0.03	**0.04**
Example 7.8	0.3		0.06	0.04	0.04	0.05	**0.04**
[Plu95, Example 3.8]	0.17	0.19	0.54	0.20	0.20	0.21	**0.21**
[Plu18, Figure 10]	0.43	0.42	0.49	0.46	0.51	0.48	**0.52**
[Plu18, Example 3]	0.11	0.09	0.26	0.1	0.12	0.11	**0.12**
[Plu18, Example 6]							
[BKNZ15, Example 4]	0.30	0.35	0.56	0.24	0.26	0.31	**0.27**
[BKNZ15, Example 5]			1.11		0.89	0.82	**0.83**
[BKNZ15, Example 6]			2.44		1.20	1.25	**1.34**
[BKZ14, Example 1]	0.10		0.18	0.09	0.11	0.19	**0.11**
[BKZ14, Example 4]	0.07	0.08	0.26	0.09	0.08	0.10	**0.10**
[BKZ14, Example 5]	0.61	0.50	2.39	0.49	0.68	0.57	**0.52**
[BKZ14, Example 6]				0.23		0.47	**0.39**

TABLE 1. Benchmark for the examples from [BKZ14, BKNZ15, Plu95, Plu18, OE24b] and the examples from Section 7. The columns display different semiring configurations: A, T and N stand for arctic, tropical and arithmetic semiring, respectively. Runtimes are indicated in seconds (s). Observe that [BKZ14, Example 6] requires either a combination of both arctic with tropical, or tropical with arithmetic semiring, for different iterations of the relative termination proof.

semiring.[6] The techniques proposed in the current paper successfully establish termination of all examples, with the exception of [Plu18, Example 6]. See Example 7.9 for a discussion about this system.

9. Related Work

The approaches [BKZ14, BKNZ15] by Bruggink et al. are the most relevant to our paper. Both employ weighted type graphs for multigraphs to prove termination of graph transformation. We significantly strengthen the weighted type graphs technique for rewriting with monic matching and we generalize the techniques to arbitrary categories and different variants of DPO.

Plump has proposed two systematic termination criteria [Plu95, Plu18] for hypergraph rewriting with DPO. The approach in [Plu95] is based on the concept of forward closures.

also not included [Plu95, Example 4.1] since this concerns a category of term graphs (which our tool does not support). The paper [OE24b] contains examples of terminating PBPO$^+$ systems. We have included those examples that have DPO equivalents; see Example 7.3 and Example 7.4.

[6]The benchmarks were obtained on a laptop with an i7-8550U cpu having 4 cores, a base clock of 1.8GHz and boost of 4GHz.

The paper [Plu18] introduces a technique for modular termination proofs based on sequential critical pairs. The modular approach can often simplify the termination arguments significantly. However, the smaller decomposed systems still require termination arguments. Therefore, our techniques and the approach from [Plu18] complement each other perfectly. Our technique can handle all examples from the papers [Plu95, Plu18], except for one (see Example 7.9 for discussion of this system).

In [OE24b], we have proposed a termination approach based on weighted element counting. Like the method presented in this paper, the method [OE24b] is defined for general categories (more specifically, rm-adhesive quasitoposes), but for PBPO$^+$ [OER23, Ove24] instead of DPO. PBPO$^+$ subsumes DPO in the quasitopos setting [OER23, Theorem 72] if both the left-hand morphism l of DPO rules and matches m are required to be regular monic. The weighted element counting approach defines the weight of an object G by means of counting weighted elements of the form $t : T \to G$, meaning it is roughly dual to the approach presented in this paper.

An example that the weighted type graph approach can prove, but the weighted element counting approach cannot prove, is Example 7.7 [BKNZ15, Example 6], which requires measuring a global property rather than a local one. Conversely, Example 7.9 [Plu18, Example 6] is an example that seems unprovable using weighted type graphs, while the element counting approach can prove it rather straightforwardly [OE24b, Example 5.9]. So it appears the two methods may be complementary even in settings where they are both defined.

Levendovszky et al. [LPE06] propose a criterion for termination of DPO rules with respect to injective matching. Roughly speaking, they show that a system is terminating if the size of repeated sequential compositions of rules tends to infinity. It remains to be seen if this criterion can be fruitfully automated.

Bottoni et al. [BHPT05] introduce a framework for proving termination of high-level replacement units (these are systems with external control expressions). They show that their framework can be used for node and edge counting arguments which are subsumed by weighted type graphs (see further [BKNZ15]).

There are also various techniques that generalize TRS methods to term graphs [Plu99, Plu97, MS16] and drags [DJ18].

Remark 9.1. The current paper is an extension of [EO24b] with full proofs, additional examples, and a gentle introduction to our technique (in Section 2). Additionally, we repair a mistake in Definition 5.2 concerning the definition of strong traceability.

10. Future Work

There are several interesting directions for future research:

(1) Can the weighted type graphs technique be extended to other algebraic graph transformation frameworks, such as SPO [Löw93], SqPO [CHHK06], PBPO [CDE$^+$19], AGREE [CDE$^+$20] and PBPO$^+$ [OER23]?
(2) Can the technique be enhanced to deal with examples like Example 7.9? (See also Open Question 7.10.)
(3) Can the technique be extended to deal with negative application conditions [HHT96]?
(4) Are there semirings other than arctic, tropical and arithmetic that can be fruitfully employed for termination proofs with the weighted type graph technique?

Concerning (1), an extension to PBPO$^+$ would be particularly interesting. In [OER23], it has been shown that, in the setting of quasitoposes, every rule of the other formalisms can be straightforwardly encoded as a PBPO$^+$ rule that generates the same rewrite relation.[7] However, our proofs (Lemma 5.13) crucially depend on a property specific to pushouts (Lemma 5.12). It will therefore be challenging to lift the reasoning to the pullbacks in PBPO$^+$ without (heavily) restricting the framework (to those pullbacks that are pushouts).

References

[AHS06] J. Adámek, H. Herrlich, and G. E. Strecker. Abstract and concrete categories: The joy of cats. 17:1–507, 2006. URL: http://www.tac.mta.ca/tac/reprints/articles/17/tr17.pdf.

[AP93] K. R. Apt and D. Pedreschi. Reasoning about termination of pure prolog programs. *Inf. Comput.*, 106(1):109–157, 1993. doi:10.1006/inco.1993.1051.

[Bey22] D. Beyer. Progress on software verification: SV-COMP 2022. In *Proc. TACAS (2)*, LNCS 13244, pages 375–402. Springer, 2022. doi:10.1007/978-3-030-99527-0_20.

[BGS11] B. Braatz, U. Golas, and T. Soboll. How to delete categorically - two pushout complement constructions. *J. Symb. Comput.*, 46(3):246–271, 2011. doi:10.1016/j.jsc.2010.09.007.

[BHK21] N. Behr, R. Harmer, and J. Krivine. Concurrency theorems for non-linear rewriting theories. In *Proc. Conf. on Graph Transformation (ICGT)*, volume 12741 of *LNCS*, pages 3–21. Springer, 2021. doi:10.1007/978-3-030-78946-6_1.

[BHK22] N. Behr, R. Harmer, and J. Krivine. Fundamentals of compositional rewriting theory. *CoRR*, abs/2204.07175, 2022. arXiv:2204.07175, doi:10.48550/arXiv.2204.07175.

[BHPT05] P. Bottoni, K. Hoffmann, F. Parisi-Presicce, and G. Taentzer. High-level replacement units and their termination properties. *J. Vis. Lang. Comput.*, 16(6):485–507, 2005. doi:10.1016/j.jvlc.2005.07.001.

[BKNZ15] H. J. S. Bruggink, B. König, D. Nolte, and H. Zantema. Proving termination of graph transformation systems using weighted type graphs over semirings. In *Proc. Conf. on Graph Transformation (ICGT)*, volume 9151 of *LNCS*, pages 52–68. Springer, 2015. doi:10.1007/978-3-319-21145-9_4.

[BKNZ23] H. J. S. Bruggink, B. König, D. Nolte, and H. Zantema. Proving termination of graph transformation systems using weighted type graphs over semirings, 2023. arXiv:1505.01695v3. arXiv:1505.01695.

[BKZ14] H. J. S. Bruggink, B. König, and H. Zantema. Termination analysis for graph transformation systems. In *Proc. Conf. on Theoretical Computer Science (TCS)*, volume 8705 of *LNCS*, pages 179–194. Springer, 2014. doi:10.1007/978-3-662-44602-7_15.

[BPHF22] K. Brown, E. Patterson, T. Hanks, and J. P. Fairbanks. Computational category-theoretic rewriting. In *Proc. Conf. on Graph Transformation (ICGT)*, volume 13349 of *LNCS*, pages 155–172. Springer, 2022. doi:10.1007/978-3-031-09843-7_9.

[CDE+19] A. Corradini, D. Duval, R. Echahed, F. Prost, and L. Ribeiro. The PBPO graph transformation approach. *J. Log. Algebraic Methods Program.*, 103:213–231, 2019.

[CDE+20] A. Corradini, D. Duval, R. Echahed, F. Prost, and L. Ribeiro. Algebraic graph rewriting with controlled embedding. *Theor. Comput. Sci.*, 802:19–37, 2020. doi:10.1016/j.tcs.2019.06.004.

[CHHK06] A. Corradini, T. Heindel, F. Hermann, and B. König. Sesqui-Pushout Rewriting. In *Proc. Conf. on Graph Transformation (ICGT)*, volume 4178 of *LNCS*, pages 30–45. Springer, 2006. doi:10.1007/11841883_4.

[CMR+97] A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe. Algebraic approaches to graph transformation - part I: basic concepts and double pushout approach. In *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations*, pages 163–246. World Scientific, 1997.

[7] The subsumption of SPO by PBPO$^+$ in quasitoposes remains a conjecture [OER23, Remark 23], but has been established for **Graph**.

[CPR06] B. Cook, A. Podelski, and A. Rybalchenko. Termination proofs for systems code. In *Proc. Conf. on Programming Language Design and Implementation*, pages 415–426. ACM, 2006. doi:10.1145/1133981.1134029.

[CPR11] B. Cook, A. Podelski, and A. Rybalchenko. Proving program termination. *Commun. ACM*, 54(5):88–98, 2011. doi:10.1145/1941487.1941509.

[DJ18] N. Dershowitz and J.-P. Jouannaud. Graph path orderings. In *Proc. Conf. on Logic for Programming, Artificial Intelligence and Reasoning (LPAR)*, volume 57 of *EPiC Series in Computing*, pages 307–325. EasyChair, 2018. doi:10.29007/6hkk.

[dMB08] L. M. de Moura and N. S. Bjørner. Z3: an efficient SMT solver. In *Proc. Conf. on on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 4963 of *LNCS*, pages 337–340. Springer, 2008. doi:10.1007/978-3-540-78800-3_24.

[EEPT06] H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. *Fundamentals of Algebraic Graph Transformation*. Monographs in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-31188-2.

[EGH10] H. Ehrig, U. Golas, and F. Hermann. Categorical frameworks for graph transformation and HLR systems based on the DPO approach. *Bull. EATCS*, 102:111–121, 2010.

[EO24a] J. Endrullis and R. Overbeek. Generalized weighted type graphs for termination of graph transformation systems, 2024. URL: https://arxiv.org/abs/2307.07601, arXiv:2307.07601.

[EO24b] J. Endrullis and R. Overbeek. Generalized weighted type graphs for termination of graph transformation systems. In *Proc. Conf. on Graph Transformation (ICGT)*, volume 14774 of *LNCS*, pages 39–58. Springer, 2024. doi:10.1007/978-3-031-64285-2_3.

[EPS73] H. Ehrig, M. Pfender, and H. J. Schneider. Graph-Grammars: An Algebraic Approach. In *Proc. Symp. on on Switching and Automata Theory (SWAT)*, page 167–180. IEEE Computer Society, 1973. doi:10.1109/SWAT.1973.11.

[GBE+14] J. Giesl, M. Brockschmidt, F. Emmes, F. Frohn, C. Fuhs, C. Otto, M. Plücker, P. Schneider-Kamp, T. Ströder, S. Swiderski, and R. Thiemann. Proving termination of programs automatically with AProVE. In *Proc. Conf. on Automated Reasoning (IJCAR)*, volume 8562 of *LNCS*, pages 184–191. Springer, 2014. doi:10.1007/978-3-319-08587-6_13.

[GRS+11] J. Giesl, M. Raffelsieper, P. Schneider-Kamp, S. Swiderski, and R. Thiemann. Automated termination proofs for haskell by term rewriting. *ACM Trans. Program. Lang. Syst.*, 33(2):7:1–7:39, 2011. doi:10.1145/1890028.1890030.

[GRS+19] J. Giesl, A. Rubio, C. Sternagel, J. Waldmann, and A. Yamada. The termination and complexity competition. In *Proc. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 11429 of *LNCS*, pages 156–166. Springer, 2019. doi:10.1007/978-3-030-17502-3_10.

[HHT96] A. Habel, R. Heckel, and G. Taentzer. Graph grammars with negative application conditions. *Fundam. Informaticae*, 26(3/4):287–313, 1996. doi:10.3233/FI-1996-263404.

[HMP01] A. Habel, J. Müller, and D. Plump. Double-pushout graph transformation revisited. *Math. Struct. Comput. Sci.*, 11(5):637–688, 2001. doi:10.1017/S0960129501003425.

[JLS07] P. T. Johnstone, S. Lack, and P. Sobocinski. Quasitoposes, quasiadhesive categories and Artin glueing. In *Algebra and Coalgebra in Computer Science (CALCO)*, volume 4624 of *LNCS*, pages 312–326. Springer, 2007. doi:10.1007/978-3-540-73859-6_21.

[KNPR18] B. König, D. Nolte, J. Padberg, and A. Rensink. A tutorial on graph transformation. In *Graph Transformation, Specifications, and Nets - In Memory of Hartmut Ehrig*, volume 10800 of *LNCS*, pages 83–104. Springer, 2018. doi:10.1007/978-3-319-75396-6_5.

[LJB01] C. S. Lee, N. D. Jones, and A. M. Ben-Amram. The size-change principle for program termination. In *Proc. Symp. on Principles of Programming Languages (POPL)*, pages 81–92. ACM, 2001. doi:10.1145/360204.360210.

[Löw93] M. Löwe. Algebraic approach to single-pushout graph transformation. *Theor. Comput. Sci.*, 109(1&2):181–224, 1993. doi:10.1016/0304-3975(93)90068-5.

[LPE06] T. Levendovszky, U. Prange, and H. Ehrig. Termination criteria for DPO transformations with injective matches. In *Proc. Workshop on Graph Transformation for Concurrency and Verification*, volume 175 of *ENTCS*, pages 87–100. Elsevier, 2006. doi:10.1016/J.ENTCS.2007.04.019.

[LS04] S. Lack and P. Sobociński. Adhesive categories. In *Proc. Conf. on Foundations of Software Science and Computation Structures (FOSSACS)*, volume 2987 of *LNCS*, pages 273–288. Springer, 2004. doi:10.1007/978-3-540-24727-2_20.

[MS16] G. Moser and M. A. Schett. Kruskal's tree theorem for acyclic term graphs. In *Proc. Workshop on Computing with Terms and Graphs, TERMGRAPH*, volume 225 of *EPTCS*, pages 25–34, 2016. doi:10.4204/EPTCS.225.5.

[OE24a] R. Overbeek and J. Endrullis. graphTT: A termination tool for algebraic graph transformation, September 2024. doi:10.5281/zenodo.13842805.

[OE24b] R. Overbeek and J. Endrullis. Termination of graph transformation systems using weighted subgraph counting. *Log. Methods Comput. Sci.*, Volume 20, Issue 4, Nov 2024. doi:10.46298/lmcs-20(4:12)2024.

[OER23] R. Overbeek, J. Endrullis, and A. Rosset. Graph rewriting and relabeling with PBPO$^+$: A unifying theory for quasitoposes. *Journal of Logical and Algebraic Methods in Programming*, 2023. doi:10.1016/j.jlamp.2023.100873.

[Ove24] R. Overbeek. *A Unifying Theory for Graph Transformation*. PhD thesis, Vrije Universiteit Amsterdam, 2024. doi:10.5463/thesis.524.

[Plu95] D. Plump. On termination of graph rewriting. In *Proc. Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 1017 of *LNCS*, pages 88–100. Springer, 1995. doi:10.1007/3-540-60618-1_68.

[Plu97] D. Plump. Simplification orders for term graph rewriting. In *Proc. Symp. on Mathematical Foundations of Computer Science (MFCS)*, volume 1295 of *LNCS*, pages 458–467. Springer, 1997. doi:10.1007/BFb0029989.

[Plu99] D. Plump. Term graph rewriting. *Handbook Of Graph Grammars And Computing By Graph Transformation: Volume 2: Applications, Languages and Tools*, pages 3–61, 1999. URL: https://www-users.york.ac.uk/~djp10/Papers/tgr_survey.pdf.

[Plu18] D. Plump. Modular termination of graph transformation. In *Graph Transformation, Specifications, and Nets*, volume 10800 of *LNCS*, pages 231–244. Springer, 2018. doi:10.1007/978-3-319-75396-6_13.

[Vig03] Sebastiano Vigna. A guided tour in the topos of graphs, 2003. doi:10.48550/arxiv.math/0306394.

[Zan03] H. Zantema. Termination. In *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*, pages 181–259. Cambridge University Press, 2003.