A 2-Categorical Presentation of Term Graph Rewriting*

A. Corradini[1], F. Gadducci[2,3]

[1] CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands, (andrea@cwi.nl).
[2] TUB, Fachbereich 13 Informatik, Franklinstraße 28/29, 10587 Berlin, Germany, (gfabio@cs.tu-berlin.de).
[3] Corresponding author: Phone: +49-30-31473557, Fax: +49-30-31423516.

Abstract. It is well-known that a term rewriting system can be faithfully described by a cartesian 2-category, where horizontal arrows represent terms, and cells represent rewriting sequences. In this paper we propose a similar, original 2-categorical presentation for *term graph rewriting*. Building on a result presented in [8], which shows that term graphs over a given signature are in one-to-one correspondence with arrows of a gs-monoidal category freely generated from the signature, we associate with a term graph rewriting system a gs-monoidal 2-category, and show that cells faithfully represent its rewriting sequences. We exploit the categorical framework to relate term graph rewriting and term rewriting, since gs-monoidal (2-)categories can be regarded as "weak" cartesian (2-)categories, where certain (2-)naturality axioms have been dropped.

Keywords: term graph rewriting, directed acyclic graphs, categorical models, 2-categories.

1 Introduction

The classical theory of *term graph rewriting* studies the issue of representing finite terms as directed, acyclic graphs, and of modeling term rewriting via graph rewriting (among the many contributions to this theory, we mention [28, 2, 13, 10, 1]; see also the book [27] and the references therein). The main advantage of using graphs is that the sharing of common subterms can be represented explicitly, avoiding to copy a subterm when applying a rewrite rule with more than one occurrence of a variable in its right-hand side. Furthermore the rewriting process is speeded up, because the rewriting steps do not have to be repeated for each copy of an identical subterm. For these reasons term graph rewriting is often used, for example, in the implementation of functional languages [24].

* Research partly supported by the EC TMR Network GETGRATS (General Theory of Graph Transformation Systems) through the Technical University of Berlin. A. Corradini is on leave from Dipartimento di Informatica, Pisa, Italy, and he is also partly supported by the EC Fixed Contribution Contract n. EBRFMBICT960840.

The relationship between classical term rewriting and term graph rewriting has been studied in many papers in the literature, including those mentioned above. All the authors agree that term graph rewriting is a sort of term rewriting with explicit sharing of subterms. However, in our opinion, there is an unsatisfactory gap between the achievements of the theory of term rewriting and of term graph rewriting, respectively. On the one hand, indeed, the terms over a signature Σ have an elegant characterization, using Universal Algebra techniques, as elements of the free algebra generated by a signature, or also, using more categorical tools, as the arrows of a suitable cartesian category; and term rewriting sequences over a given system can be defined not only in the usual "operational" way, using the notions of redex and substitution, but also more abstractly as the cells of a suitable cartesian 2-category. Such a categorical account has the advantage of being independent of representation details, and of stressing the intrinsic algebraic structure of terms and their rewriting; in fact, on the basis of the mentioned results one can safely claim that the essential structure of the collection of terms over a signature is "cartesianity".

On the other hand, only more concrete and operational presentations of term graphs and their rewriting are available in literature. For example, term graphs have been defined, among other ways, as directed (acyclic) graphs satisfying a number of constraints [2], as suitable labeled hypergraphs called *jungles* [13], or as sets of recursive equations [1]. None of these (essentially equivalent) representations is able to highlight the algebraic structure of term graphs, and a consequence of this is that the simple correspondence they have with terms is often made more obscure by the need of encoding and decoding functions. In fact, for each of the mentioned representations of term graphs suitable functions have been defined that, for example, extract from a term graph the term it represents (via some "unraveling"), or, viceversa, that from a given term return a term graph representing it (and exhibiting some minimal or maximal sharing).

The goal of this paper (together with its predecessor [8]) is to fill the gap described above, by proposing an original, clean categorical description of term graph rewriting. More precisely, while [8] has shown that term graphs over a signature correspond one-to-one to arrows of a *gs-monoidal* category, building on that result we show here that term graph rewriting sequences (satisfying a mild restriction) are faithfully represented by cells of a gs-monoidal 2-category. Summing up, one could say that "gs-monoidality characterizes term graph rewriting", in the same sense as "cartesianity characterizes term rewriting".

Before explaining concisely the main contribution of the paper, it is worth recalling that the terms over a given signature Σ can be regarded as the arrows of a cartesian category (called the *algebraic theory* of Σ) freely generated (in a suitable way) by Σ (see e.g. [21, 18]). Such a category has (underlined) natural numbers as objects, and its generators are arrows like $g : \underline{n} \to \underline{1}$, where g is an operator of arity n in Σ; in this category arrows from $\underline{n}$ to $\underline{m}$ are in one-to-one correspondence with m-tuples of terms over n variables. Furthermore, a term rewrite rule $R = l \to r$ (where l and r are two terms over Σ) can be represented as a *cell*, i.e., a vertical arrow from the arrow representing l to the arrow representing r. This situation can be denoted by $R : l \Rightarrow r : \underline{n} \to \underline{1}$, which also states that l and r are arrows from $\underline{n}$ to $\underline{1}$, i.e., that they have at most n variables. Given a term rewriting system, the structure obtained in this way, i.e., the algebraic theory of Σ enriched with one cell for each rule, is called a *computad* [30]; from such a computad, a free construction can generate a (cartesian) 2-category by adding all identity cells, and closing cells with respect to horizontal and vertical composition. The interesting fact is that the resulting 2-category faithfully represents all the possible rewriting sequences of the original system [26, 25, 23, 30]. In fact, horizontal composition of cells generates all the possible instantiations of the rules and, at the same time, places rules in all possible contexts. Vertical composition acts instead as sequential composition.

Cartesian (2-)categories can be defined as *symmetric monoidal (2-)categories* equipped with two *(2-)natural (symmetric monoidal) (2-)transformations* denoted by ∇ (the *duplicator*) and ! (the *discharger*), respectively. The definition of cartesian (2-)categories can be weakened slightly by dropping the requirement of (2-)naturality for ∇ and !: we call the resulting (2-)categories *gs-monoidal*. As anticipated above, the main result of [8] shows that term graphs over a given signature Σ are in one-to-one correspondence with the arrows of the *gs-monoidal theory* of Σ, i.e., a gs-monoidal category freely generated by Σ. In this paper, building on this result, after introducing a definition of term graph rewriting, we show how to generate from a term graph rewriting system a free gs-monoidal 2-category, called its *gs-monoidal 2-theory*. The main result shows that in such a 2-theory there is a cell between two term graphs if and only if they are related by a rewriting sequence where the redexes have to satisfy a mild restriction.

As expected, the categorical framework allows us to relate in a clean way term graph rewriting and term rewriting. In fact, by the above the only difference with terms is that the (2-)naturality of the duplicator and of the discharger does not hold anymore. And this fact has a clear interpretation in terms of "sharing of subterms" and of "garbage collection", in the following sense. Suppose that $g : \underline{n} \to \underline{1}$ is the arrow corresponding to a n-ary operator $g(x_1, \ldots, x_n)$. Then, using ";" for composition of arrows in diagrammatic order, arrow $a_1 = g\,;\nabla_{\underline{1}} : \underline{n} \to \underline{2}$ represents, intuitively, a structure having two pointers to $g(x_1, \ldots, x_n)$, while arrow $a_2 = \nabla_{\underline{n}}\,; g \oplus g : \underline{n} \to \underline{2}$ represents a structure where two copies of g share the same variables (Figure 1 shows the term graphs corresponding to a_1 and a_2, using conventions to be introduced in Section 2; in particular the numbers to the left represent the variables, and those to the right the roots).

Now, regarded as term graphs the two structures must be distinct, because they exhibit a different degree of sharing for some substructure: There are two "pointers" to g in G_1, but only one to each copy of g in G_2. Indeed, arrows a_1 and a_2 are distinct in the free gs-monoidal category concerned. On the contrary, in the free cartesian category, the naturality of ∇ implies that $g\,;\nabla_{\underline{1}} = \nabla_{\underline{n}}\,; g \oplus g$, which means that the two arrows are provably the same. Therefore structures with different degree of sharing are identified, and, by convention, one can take as representative of an equivalence class of such arrows the structure with less sharing, that is, a tuple of trees (in our example, arrow a_1 ($= a_2$) in the concerned

Fig. 1. Two term graphs that correspond to the same tuple of terms.

algebraic theory represents the tuple of terms $\langle g(x_1, \ldots, x_n), g(x_1, \ldots, x_n)\rangle\rangle$.

By similar arguments it can be shown that the categorical counterpart of "implicit garbage collection" is the naturality of the discharger "!". Let us call "garbage" a substructure that is not accessible by any root. Then arrow $g; !_{\underline{1}} : \underline{n} \to \underline{0}$ can be regarded as the structure $g(x_1, \ldots, x_n)$ where g is garbage, because, intuitively, its only root is deleted by the discharger $!_{\underline{1}}$. On the other hand, arrow $!_{\underline{n}} : \underline{n} \to \underline{0}$ represents an empty structure. Now the naturality of ! implies that $g; !_{\underline{1}} = !_{\underline{n}}$, i.e., that any structure with some garbage is equivalent to the same structure with the garbage removed, and we may call this "implicit garbage collection".

The paper is organized as follows. In Section 2 we introduce *ranked term graphs*, i.e., equivalence classes of directed acyclic graphs labeled over a signature Σ, with distinguished lists of *variable* and *root* nodes. Such nodes are used for defining an operation of "composition", which is the counterpart of term substitution, and that provides term graphs with a natural categorical structure. We then introduce our definition of term graph rewriting. Section 3 recalls the basic definitions about *symmetric monoidal 2-categories*, and introduces *gs-monoidal 2-categories*: Symmetric monoidal 2-categories equipped with two transformations, satisfying a few properties. In Section 4 the main result of the paper is presented: Namely, that for each term graph rewriting system, a gs-monoidal 2-category can be freely generated, such that its cells faithfully represent the rewriting sequences of the original system. In Section 5 we discuss the relationship between our notion of term graph rewriting and term rewriting. Finally Section 6 concludes suggesting further investigations, which include a comparative study of the classical definition of term graph rewriting in [2]; the analysis of the properties of the equivalence induced on rewriting sequences by their representation as cells; and lifting the results presented here to *cyclic* term graph rewriting.

2 Term Graphs and Term Graph Rewriting

This section introduces (ranked) term graphs as isomorphism classes of (ranked) directed acyclic graphs, as well as the basic definitions related to Term Graph Rewriting. Our main concern in the next sections will be to stress the underlying categorical structures, hence the following presentation slightly departs from the standard definition However, apart from the restriction to the acyclic case, we can safely see our term graphs as a minor variation of those defined in the seminal paper [2]: A full discussion of their relationship can be found in [8].

Definition 1 (directed acyclic graphs, dags). Let Σ be a signature, i.e., a ranked set of operator symbols, and let *arity* be the function returning the arity of an operator symbol, i.e., $arity(f) = n$ iff $f \in \Sigma_n$. A *labeled graph* d *(over* Σ*)* is a triple $d = \langle N, l, s \rangle$, where N is a set of *nodes* (ranged over by $a, b, \ldots$), $l : N \to \Sigma$ is a partial function called the *labeling function*, $s : N \to N^*$ is a partial function called the *successor function*, and such that the following conditions are satisfied:

- $dom(l) = dom(s)$, i.e., labeling and successor functions are defined on the same subset of N; a node $a \in N$ is called *empty* if $a \notin dom(l)$.
- for each node $a \in dom(l)$, $arity(l(a)) = length(s(a))$, i.e., each non-empty node has as many successor nodes as the arity of its label.

If $s(n) = \langle n_1, \ldots, n_k \rangle$, we say that n_i is the *i-th successor* of n and denote it by $s(n)_i$. A labeled graph is *discrete* if all its nodes are empty. A *path* in d is a sequence $\langle a_0, i_0, a_1, \ldots, i_{m-1}, a_m \rangle$, where $m \geq 0$, $a_0, \ldots, a_m \in N$, $i_0, \ldots, i_{m-1} \in \mathbb{N}$ (the natural numbers), and a_k is the i_{k-1}-th successor of a_{k-1} for $k \in \{1, \ldots, m\}$. The *length* of this path is m; if $m = 0$, the path is *empty*. A *cycle* is a path like above where $a_0 = a_m$.

A *directed acyclic graph* or *dag (over* Σ*)*, is a labeled graph which does not contain any non-empty cycle. If $a \in N$ is a node of a dag $d = \langle N, l, s \rangle$, then by $d|a$ we denote the sub-dag of d rooted at a, defined in the obvious way. For a dag d we shall often denote its components by $N(d)$, l_d and s_d, respectively. Moreover, $N_\emptyset(d)$ and $N_\Sigma(d)$ denote the set of empty and non-empty nodes of d, respectively (thus $N(d) = N_\Sigma(d) \uplus N_\emptyset(d)$, where $\uplus$ denotes disjoint union). □

Definition 2 (dag morphisms, category $\mathbf{Dag}_\Sigma$). Let d and d' be two dags. A *(dag) morphism* $f : d \to d'$ is a function $f : N(d) \to N(d')$ that preserves labeling and successors, i.e., such that for each node $a \in N_\Sigma(d)$, $l_{d'}(f(a)) = l_d(a)$, and $s_{d'}(f(a))_i = f(s_d(a)_i)$ for each $i \in \{1, \ldots, arity(l_d(a))\}$. A dag morphism $f : d \to d'$ is *Σ-injective* if its restriction to $N_\Sigma(d)$ is injective; it is *strongly acyclic* if for all $a_1 \in N_\emptyset(d)$ and $a_2 \in N_\Sigma(d)$ there is no path from $f(a_1)$ to $f(a_2)$ in d'; and it is *plain* if it is both Σ-injective and strongly acyclic. It is easy to verify that Σ-injective morphisms compose, as well as strongly acyclic ones.

Dags over Σ and dag morphisms clearly form a category that will be denoted $\mathbf{Dag}_\Sigma$. □

In the following, for each $n \in \mathbb{N}$ we shall denote by $\underline{n}$ the set $\underline{n} = \{1, \ldots, n\}$ (thus $\underline{0} = \emptyset$).

Definition 3 (ranked dags and term graphs). An *(i, j)-ranked dag* (or also, a *dag of rank* (i, j)) is a triple $g = \langle r, d, v \rangle$, where d is a dag with exactly j empty nodes, $r : \underline{i} \to N(d)$ is a function called the *root mapping*, and $v : \underline{j} \to N_\emptyset(d)$

is a bijection between $\underline{j}$ and the empty nodes of d, called the *variable mapping*. Node $r(k)$ is called the *k-th root* of d, and $v(k)$ is called the *k-th variable* of d, for each admissible k.

Two (i, j)-ranked dags $g = \langle r, d, v\rangle$ and $g' = \langle r', d', v'\rangle$ are *isomorphic* if there exists a *ranked dag isomorphism* $\phi : g \to g'$, i.e., a dag isomorphism $\phi : d \to d'$ such that $\phi \circ r = r'$ and $\phi \circ v = v'$. A *(i, j)-ranked term graph* G (or *with rank* (i, j)) is an isomorphism class of (i, j)-ranked dags. Sometimes we will write G^i_j to recall that G has rank (i, j). □

We introduce now two operations on ranked term graphs. The *composition* of two ranked term graphs is obtained by gluing the variables of the first one with the roots of the second one, and it is defined only if their number is equal. This operation allows us to define a category having ranked term graphs as arrow. Next the *union* of term graphs is introduced: it is always defined, and it is a sort of disjoint union where roots and variables are suitably renumbered. This second operation provides the category of term graphs with a monoidal structure that will be made explicit in the next sections.

Definition 4 (composition of ranked term graphs). Let ${G'}^j_k = [\langle r', d', v'\rangle]$ and $G^i_j = [\langle r, d, v\rangle]$ be two ranked term graphs. Their *composition* is the ranked term graph $H^i_k = {G'}^j_k ; G^i_j$ defined as $H^i_k = [\langle in_d \circ r, d'', in_{d'} \circ v'\rangle]$, where $d'', in_d : d \to d''$ and $in_{d'} : d' \to d''$ are obtained as follows. Assuming that $d = \langle N(d), l_d, s_d\rangle$ and $d' = \langle N(d'), l_{d'}, s_{d'}\rangle$, we have $d'' = \langle (N(d) \uplus N(d'))/_{\approx}, l'', s''\rangle$, where $\approx$ is the least equivalence relation such that $v(i) \approx r'(i)$ for $i \in \underline{j}$, and l'' and s'' are determined by l_d and s_d, respectively, for all $\approx$-equivalence classes containing only nodes of d, and by $l_{d'}$ and $s_{d'}$, respectively, for all other classes. Furthermore, the injections $in_d : d \to d''$ and $in_{d'} : d' \to d''$ map each node to its $\approx$-equivalence class. □

It is easy to check that both morphisms in_d and $in_{d'}$ are plain. It is worth stressing that the composed dag can be characterized elegantly as a pushout, in the sense that $\langle d'', in_d, in_{d'}\rangle$ is a pushout of $\langle v : \underline{j} \to d, r' : \underline{j} \to d'\rangle$ in $\mathbf{Dag}_\Sigma$ (set $\underline{j}$ is regarded as a discrete dag). Thus the well-definedness of composition of term graphs easily follows from the uniqueness of pushouts up to isomorphism.[4]

Definition 5 (the category of term graphs). For a given signature Σ, $\mathbf{TG}_\Sigma$ denotes the category having as objects underlined natural numbers, and as arrows from $\underline{n}$ to $\underline{m}$ all (m, n)-ranked term graphs. Arrow composition is defined as in Definition 4, and the identity on $\underline{n}$ is the term graph $G^{\underline{n}}_{id}$ of rank (n, n) having n nodes, and where the i-th root is also the i-th variable, for all $i \in \underline{n}$. Then it is easy to check that $\mathbf{TG}_\Sigma$ is a well-defined category, because composition is associative, and the identity laws hold. □

[4] Actually, the pushout of two arrows in category $\mathbf{Dag}_\Sigma$ does not always exist: it does exist however in the case we are interested in, since morphism $v : \underline{j} \to d$ is injective and has only empty nodes in the codomain. See [10] for necessary and sufficient conditions for the existence of pushouts in the equivalent category of *jungles*.

Definition 6 (union of ranked term graphs). Let $G^i_j = [\langle r, d, v\rangle]$ and $G'^k_l = [\langle r', d', v'\rangle]$ be two ranked term graphs. Their *union* or *parallel composition* is the term graph of rank $(i + k, j + l)$ $G^i_j \oplus G'^k_l = [\langle r'', d \uplus d', v''\rangle]$, where $r'' : \underline{i+k} \to d \uplus d'$ and $v'' : \underline{j+l} \to d \uplus d'$ are defined as

– $r''(x) = \begin{cases} r(x) & \text{if } x \in \underline{i} \\ r'(x-i) & \text{if } x \in \{i+1, \ldots, i+k\}. \end{cases}$
– $v''(x) = \begin{cases} v(x) & \text{if } x \in \underline{j} \\ v'(x-i) & \text{if } x \in \{j+1, \ldots, j+l\}. \end{cases}$

□

Example 1 (term graphs, composition and union). Four term graphs are shown in Figure 2. Empty nodes are represented by the natural numbers corresponding to their position in the list of variables, and are depicted as a vertical sequence on the left; non-empty nodes are represented by their label, from where the edges pointing to the successors leave; the list of numbers on the right represent pointers to the roots: A dashed arrow from j to a node indicates that it is the j-th root. For example, the first term graph G_1 has rank $(4, 2)$, four nodes (two empty, 1 and 2, and two non-empty, f and g), the successors of g are the variables 2 and 1 (in this order), the successors of f are g and 2, and the four roots are g, f, 2, and f. These graphical conventions make easy the operation of composition, that can be performed by matching the roots of the first graph with the variables of the second one, and then by eliminating them. For example, term graph $G_1 ; G_2$ is the composition of G_1 and of G_2 of rank $(1, 4)$. The last term graph is $G_1 \oplus G_2$, the union of G_1 and G_2, of rank $(5, 6)$. □

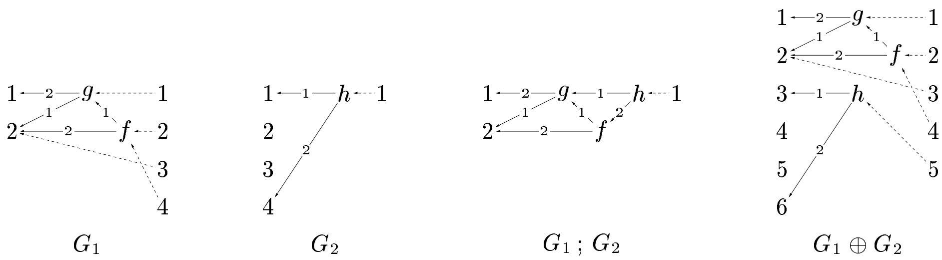

Fig. 2. Two term graphs, their composition and their union.

Definition 7 (rule, redex, term graph rewriting system). A *rule* is a triple $\mathbf{R} = \langle d, r_l, r_r\rangle$, where d is a dag, and r_l, r_r are two distinguished nodes of d, called the *left root* and the *right root*, respectively. Furthermore, we require that (1) node r_l is not empty, (2) all nodes of d are reachable from r_l or r_r, (3) all empty nodes in d are reachable from r_l, and (4) there is not path from r_r to r_l. A *redex* in a dag d_0 is a pair $\Delta = \langle \mathbf{R}, f\rangle$ where $\mathbf{R}$ is a rule as above, and $f : d|r_l \to d_0$ is a dag morphism from $d|r_l$, the sub-dag of d rooted at r_l, to d_0. Node $f(r_l)$ of d_0 is called the *root* of the redex Δ. Redex $\langle \mathbf{R}, f\rangle$ is *plain* if so is f.

A *(term graph) rewriting system* is a pair $\mathcal{R} = \langle \{\mathbf{R}_i\}_{i \in I}, \Sigma \rangle$ such that for all $i \in I$, $\mathbf{R}_i$ is a rule over Σ. $\square$

Note that condition (4) above is not required for rules in [2]. We don't allow for a non-empty path from r_r to r_l to ensure that the result of an application of the rule is an acyclic dag. And we don't allow for an empty path ($r_l = r_r$) even if we could, because such rules would result in trivial identities, making nevertheless some statements more awkward in the rest of the paper.

Definition 8 (ranked dag rewriting). Let $g_0 = \langle r_0, d_0, v_0 \rangle$ be an (i,j)-ranked dag, and let $\Delta = \langle \langle d, r_l, r_r \rangle, f : d|r_l \to d_0 \rangle$ be a redex in it. We say that g_0 *rewrites to* g_3 *via* Δ, denoted as $g_0 \to_\Delta g_3$, if g_3 is the (i,j)-ranked dag obtained through the following procedure:

[**Build phase**] Define d_1 as the dag having as nodes the disjoint union of $N(d_0)$ and $N(d|r_r) - N(d|r_l)$. Functions l_{d_1} and s_{d_1} are defined as

- $l_{d_1}(a) = \begin{cases} l_{d_0}(a) & \text{if } a \in N(d_0) \\ l_d(a) & \text{if } a \in N(d|r_r) - N(d|r_l). \end{cases}$
- $s_{d_1}(a)_i = \begin{cases} s_{d_0}(a)_i & \text{if } a \in N(d_0) \\ s_d(a)_i & \text{if } a, s_d(a)_i \in N(d|r_r) - N(d|r_l) \\ f(s_d(a)_i) & \text{if } a \in N(d|r_r) - N(d|r_l), s_d(a)_i \in N(d|r_l). \end{cases}$

Let $\hat{f} : d \to d_1$ be the obvious extension of morphism $f : d|r_l \to d_0$ such that $\hat{f}(a) = a$ for each $a \in N(d|r_r) - N(d|r_l)$.

[**Redirection phase**] Define d_2 as the dag obtained by replacing all references to the root of the redex $f(r_l)$ by references to the image of the right root r_r. More precisely, $d_2 = \langle N(d_1), l_{d_1}, s_{d_2} \rangle$, where

$s_{d_2}(a)_i = \begin{cases} s_{d_1}(a)_i & \text{if } s_{d_1}(a)_i \neq f(r_l) \\ \hat{f}(r_r) & \text{if } s_{d_1}(a)_i = f(r_l). \end{cases}$

[**Root removal phase**] Let d_3 be the dag obtained from d_2 by removing node $f(r_l)$, and let the *track function* $tr_\Delta : N(d_0) \to N(d_3)$ be defined as follows:

$tr_\Delta(a) = \begin{cases} a & \text{if } a \neq f(r_l) \\ \hat{f}(r_r) & \text{if } a = f(r_l). \end{cases}$

Then the ranked dag g_3 is defined as $\langle tr_\Delta \circ r_0, d_3, tr_\Delta \circ v_0 \rangle$. $\square$

It is easy to check that g_3 is a well-defined (i,j)-ranked dag. In fact, condition (4) of Definition 7 ensures that d_2 is acyclic and that its node $f(r_l)$ is not the successor of any node. Moreover, $tr_\Delta \circ v_0$ is a bijection between the set of empty nodes of d_3 and $\underline{j}$ by condition (3).

The definition of dag rewriting is easily extended to term graphs.

Definition 9 (term graph rewriting). Let G be a term graph, and let $\Delta = \langle \mathbf{R}, f \rangle$ be a redex in one of its elements, say $g = \langle r, d, v \rangle$. Then for each $h \in G$ and ranked dag isomorphism $\phi : g \to h$, Δ induces a redex in h, obtained by composing f with ϕ. In this situation we say that Δ is a redex in G. Furthermore, if $g \to_\Delta g'$ and $h \to_{\langle \mathbf{R}, \phi \circ f \rangle} h'$, then it is easy to check that g' and h' are isomorphic. Thus we safely extend rewriting to term graphs, writing $G \to_\Delta G'$, where $G' = [g']$. A *term graph rewriting sequence* $G_0 \to^*_{\mathcal{R}} G_n$ over a rewriting system $\mathcal{R}$ is a sequence of $n \geq 0$ rewriting steps $G_0 \to_{\Delta_1} G_1 \ldots G_{n-1} \to_{\Delta_n} G_n$, where at each step a rule of $\mathcal{R}$ is used. A rewriting sequence is *plain* if so are all redexes in it. □

3 GS-Monoidal 2-Categories

In this section we briefly present some notions about 2-categories needed in the rest of the paper. We first recall the basic definitions: For an introduction, we refer the reader to the classical work [16]. Then we introduce the original concept of *gs-monoidal 2-categories*, which will be fundamental for our characterization of term graph rewriting.

Roughly, a 2-category simply is a category $\mathbf{C}$ such that, given any two objects a, b, the hom-set $\mathbf{C}[a, b]$, i.e., the class of arrows from a to b, is a category: moreover, these *hom-categories* satisfy suitable composition properties. An arrow in $\mathbf{C}[a, b]$, a *cell*, is denoted as $\alpha : f \Rightarrow g : a \to b$ (where $f, g : a \to b$); or graphically, as

$$\begin{tikzcd} a \arrow[r, bend left, "f"] \arrow[r, bend right, "g"'] \arrow[r, phantom, "\Downarrow\alpha"] & b. \end{tikzcd}$$

Definition 10 (2-Categories). Let $\mathbf{C}$ be a category such that each hom $\mathbf{C}[a, b]$ is a category. Moreover, let us assume that for each triple a, b, c of objects there is a (horizontal) composition function $- * -$ such that, given $\alpha : f \Rightarrow h : a \to b$ and $\beta : g \Rightarrow i : b \to c$, then $\gamma = \alpha * \beta : f; g \to h; i \in \mathbf{C}[a, c]$. Graphically,

$$\begin{tikzcd} a \arrow[r, bend left, "f"] \arrow[r, bend right, "h"'] \arrow[r, phantom, "\Downarrow\alpha"] & b \arrow[r, bend left, "g"] \arrow[r, bend right, "i"'] \arrow[r, phantom, "\Downarrow\beta"] & c \end{tikzcd} \quad = \quad \begin{tikzcd} a \arrow[r, bend left, "f;g"] \arrow[r, bend right, "h;i"'] \arrow[r, phantom, "\Downarrow\gamma"] & c \end{tikzcd} \quad \in \quad \mathbf{C}[a, c]$$

where $-;-$ denotes composition inside $\mathbf{C}$. Let us consider the following cells:

$$\begin{tikzcd} a \arrow[r, bend left, "f"] \arrow[r, bend right, "g"'] \arrow[r, phantom, "\Downarrow\alpha"] & b \arrow[r, bend left, "h"] \arrow[r, bend right, "i"'] \arrow[r, phantom, "\Downarrow\beta"] & c \arrow[r, bend left] \arrow[r, bend right] \arrow[r, phantom, "\Downarrow\gamma"] & d. \end{tikzcd}$$

A *2-category* $\underline{\mathbf{C}}$ is a category $\mathbf{C}$ (called the *underlying* category) with the structure above defined, such that the composition functions satisfy the equations:

(1) $id_a * \alpha = \alpha = \alpha * id_b$; (2) $(\alpha * \beta) * \gamma = \alpha * (\beta * \gamma)$;

(3) $f * h = f; h$ (4) $(\alpha * h) \cdot (g * \beta) = \alpha * \beta = (f * \beta) \cdot (\alpha * i)$.

where $- \cdot -$ denotes the (vertical) composition insides hom-categories.[5] □

[5] For sake of readability, since no confusion can arise, we often denote the cell corresponding to the identity of an arrow by the arrow itself. For example, axiom (1) reads as $id_{id_a} * \alpha = \alpha = \alpha * id_{id_b}$; axiom (3) reads as $id_f * id_h = id_{f;h}$; and so on.

Axioms (1) and (2) assure that there exists also an underlying horizontal category, with the same objects as $\mathbf{C}$ and cells as arrows, and (3)-(4) express the functoriality of each composition in the other. In fact, 2-categories could also be described as suitable *internal categories* in **Cat** (see [3]), that is, categories such that both its classes of arrows and of objects are categories: The remaining definitions are actually biased towards this "internal" approach.

Definition 11 (2-Functors and (2-Natural) 2-Transformations). Let $\underline{\mathbf{C}}$ and $\underline{\mathbf{D}}$ be two 2-categories. A *2-functor* $F : \underline{\mathbf{C}} \to \underline{\mathbf{D}}$ is a triple $\langle F_O, F_A, F_C \rangle$ of functions, mapping objects to objects, arrows to arrows and cells to cells, respectively, preserving identities and compositions of all kinds.

Let $F, G : \underline{\mathbf{C}} \to \underline{\mathbf{D}}$ be two parallel 2-functors: a *2-transformation* $\eta : F \Rightarrow G$ is a family of arrows of $\mathbf{D}$ indexed by objects of $\mathbf{C}$, $\eta = \{\eta_a : F_O(a) \to G_O(a) \mid a \in |\mathbf{C}|\}$; it is *2-natural* if moreover, for every cell $\alpha : f \Rightarrow g : a \to b \in \underline{\mathbf{C}}$, it verifies $F_C(\alpha) * \eta_b = \eta_a * G_C(\alpha)$. □

The paradigmatic example of 2-category—and one whose use is widespread in Computer Science—is $\underline{\mathbf{Cat}}$, having small categories as objects, functors as arrows, and natural transformations as cells. As far as rewriting is concerned, it is well-known that a suitable class of 2-categories, *algebraic 2-theories*, can be used for describing term rewriting (see e.g. [26, 25, 23, 30], and the more recent [29, 9]). As anticipated in the Introduction, we want to provide a similar categorical model for term graph rewriting: To this aim we introduce now *gs-monoidal 2-categories*—roughly, symmetric monoidal 2-categories equipped with two 2-transformations.

In particular, since the focus of the paper is the correspondence result for term graph rewriting (see Section 4), our presentation is tailored to the need of our main theorem. As a start, all our categorical constructions are "strict", in the usual sense of e.g. *strict monoidal*. Accordingly, we only consider monoidal 2-functors that preserve monoidal 2-product and 2-unit "on the nose". Moreover, for the sake of brevity, we present some definitions directly at the 2-categorical level: The analogous constructions for (normal) categories can be obtained in principle by restricting the attention to the underlying category (since each category can be seen as a 2-category whose only cells are identities).

Definition 12 (symmetric monoidal 2-categories). A *monoidal 2-category* $\underline{\mathbf{C}}$ is a triple $\langle \underline{\mathbf{C}}_0, \otimes, e \rangle$, where $\underline{\mathbf{C}}_0$ is a 2-category, $e \in |\mathbf{C}_0|$ is a distinguished object and $\otimes : \underline{\mathbf{C}}_0 \times \underline{\mathbf{C}}_0 \to \underline{\mathbf{C}}_0$ is a 2-functor, satisfying the axioms: $(\alpha \otimes \beta) \otimes \gamma = \alpha \otimes (\beta \otimes \gamma)$ and $\alpha \otimes id_e = id_e \otimes \alpha = \alpha$ for all cells $\alpha, \beta, \gamma \in \underline{\mathbf{C}}_0$.

A *symmetric monoidal 2-category* $\underline{\mathbf{C}}$ is a four-tuple $\langle \underline{\mathbf{C}}_0, \otimes, e, \rho \rangle$ where $\langle \underline{\mathbf{C}}_0, \otimes, e \rangle$ is a monoidal 2-category, and $\rho : \otimes \Rightarrow \otimes \circ \mathrm{X} : \underline{\mathbf{C}}_0 \times \underline{\mathbf{C}}_0 \to \underline{\mathbf{C}}_0$ is a 2-natural 2-transformation (where X is the 2-functor that swaps its two arguments) such that $\rho_{e,a} = id_a$ for all $a \in |\mathbf{C}_0|$, and satisfying:

$$
\begin{tikzcd}
a \otimes b \otimes c \arrow[r, "id_a \otimes \rho_{b,c}"] \arrow[dr, "\rho_{a \otimes b, c}"'] & a \otimes c \otimes b \arrow[d, "\rho_{a,c} \otimes id_b"] \\
& c \otimes a \otimes b
\end{tikzcd}
\qquad
\begin{tikzcd}
a \otimes b \arrow[r, "\rho_{a,b}"] \arrow[dr, "id_{a \otimes b}"'] & b \otimes a \arrow[d, "\rho_{b,a}"] \\
& a \otimes b
\end{tikzcd}
$$

A *monoidal 2-functor* $F : \underline{\mathbf{C}} \to \underline{\mathbf{C}}'$ is a 2-functor $F : \underline{\mathbf{C}}_0 \to \underline{\mathbf{C}}'_0$ such that $F(e) = e'$ and $F(\alpha \otimes \beta) = F(\alpha) \otimes' F(\beta)$ for all $\alpha, \beta \in \underline{\mathbf{C}}_0$. It is *symmetric* if moreover $F(\rho_{a,b}) = \rho'_{F(a),F(b)}$. □

If we restrict our attention from the symmetric monoidal 2-category $\underline{\mathbf{C}} = \langle \underline{\mathbf{C}}_0, \otimes, e, \rho \rangle$ to the underlying category $\mathbf{C}_0$ and the derived operators $\otimes_r, \rho_r$ (defined in the obvious way), it turns out that the category $\mathbf{C} = \langle \mathbf{C}_0, \otimes_r, e, \rho_r \rangle$ is symmetric (strict) monoidal, according to the classical definition (see e.g. [22]).

Definition 13 (gs-monoidal 2-categories). A *gs-monoidal 2-category* $\underline{\mathbf{C}}$ is a six-tuple $\langle \underline{\mathbf{C}}_0, \otimes, e, \rho, \nabla, ! \rangle$, where $\langle \underline{\mathbf{C}}_0, \otimes, e, \rho \rangle$ is a symmetric monoidal 2-category and $! : Id_{\underline{\mathbf{C}}_0} \Rightarrow e : \underline{\mathbf{C}}_0 \to \underline{\mathbf{C}}_0$, $\nabla : Id_{\underline{\mathbf{C}}_0} \Rightarrow \otimes \circ \Delta : \underline{\mathbf{C}}_0 \to \underline{\mathbf{C}}_0$ are two 2-transformations (Δ is the diagonal 2-functor), such that $!_e = \nabla_e = id_e$ and satisfying:

$$
\begin{tikzcd}
a \arrow[r, "\nabla_a"] \arrow[d, "\nabla_a"'] & a \otimes a \arrow[d, "id_a \otimes \nabla_a"] \\
a \otimes a \arrow[r, "\nabla_a \otimes id_a"'] & a \otimes a \otimes a
\end{tikzcd}
\qquad
\begin{tikzcd}
a \arrow[r, "\nabla_a"] \arrow[dr, "id_a"'] & a \otimes a \arrow[d, "id_a \otimes !_a"] \\
& a = a \otimes e
\end{tikzcd}
\qquad
\begin{tikzcd}
a \arrow[r, "\nabla_a"] \arrow[dr, "\nabla_a"'] & a \otimes a \arrow[d, "\rho_{a,a}"] \\
& a \otimes a
\end{tikzcd}
$$

$$
\begin{tikzcd}
a \otimes b \arrow[r, "\nabla_{a \otimes b}"] \arrow[dr, "\nabla_a \otimes \nabla_b"'] & a \otimes b \otimes a \otimes b \arrow[d, "id_a \otimes \rho_{b,a} \otimes id_b"] \\
& a \otimes a \otimes b \otimes b
\end{tikzcd}
\qquad
\begin{tikzcd}
e \arrow[dr, "id_e"'] & a \otimes b \arrow[l, "!_{a \otimes b}"'] \arrow[d, "!_a \otimes !_b"] \\
& e = e \otimes e
\end{tikzcd}
$$

A *gs-monoidal 2-functor* $F : \underline{\mathbf{C}} \to \underline{\mathbf{C}}'$ is a symmetric monoidal 2-functor such that $F(!_a) = !'_{F(a)}$ and $F(\nabla_a) = \nabla'_{F(a)}$. The category of small gs-monoidal 2-categories and gs-monoidal 2-functors is denoted by **GSM-2Cat**. □

Again, if a 2-category $\underline{\mathbf{C}}$ is gs-monoidal, also the underlying category $\mathbf{C}$ has such a structure: We denote by **GSM-Cat** the category of small *gs-monoidal categories* and *gs-monoidal functors*. Next we recall from [8] how it is possible to generate from a signature a free gs-monoidal category, called its *gs-monoidal theory.*

Definition 14 (graphs with pairing and signature graphs). A *graph with pairing* G is a six-tuple $\langle N, \times, e, A, s, t \rangle$, where $\langle N, \times, e \rangle$ is a monoid of nodes, A a set of arcs, and $s, t : A \to N$ are (the source and target) functions. A morphism of graphs with pairing $\langle f_N, f_A \rangle : G \to G'$ is a pair where $f_N : \langle N, \times, e \rangle \to \langle N', \times', e' \rangle$ is a monoid homomorphism and $f_A : A \to A'$ is a function such that sources and targets of arcs are preserved in the expected way. The category of graphs with pairing and their morphisms is denoted by **M-Gr**.

Let Σ be a one-sorted signature. By G_Σ we denote the graph with pairing $\langle \underline{\mathbb{N}}, \otimes, \underline{0}, OP_\Sigma, s, t \rangle$, where $\langle \underline{\mathbb{N}}, \otimes, \underline{0} \rangle$ is the monoid of underlined natural numbers (with $\underline{n} \otimes \underline{m} = \underline{n+m}$), $OP_\Sigma = \{f_\Sigma \mid f \in \Sigma_n \text{ for some } n \in \mathbb{N}\}$, and source and target are determined as $f_\Sigma : \underline{n} \to \underline{1}$ iff $f \in \Sigma_n$, for all $f_\Sigma \in OP_\Sigma$ (without loss of generality, we assume that the sets Σ_n are disjoint). □

Proposition 15 (gs-monoidal theories). *Let V : **GSM-Cat** $\to$ **M-Gr** be the forgetful functor mapping a gs-monoidal category to the underlying graph with pairing: It admits a left (free) adjoint GSM :* **M-Gr** $\to$ **GSM-Cat**.

The gs-monoidal theory **GS-Th***(Σ) of a given (one-sorted) signature Σ, is defined as $GSM(G_\Sigma)$, i.e., the free gs-monoidal category generated by the graph with pairing associated with Σ.* □

The main result of [8], which we recall in Theorem 18, states the existence of a one-to-one correspondence between arrows of the hom-set $\mathbf{GS\text{-}Th}(\Sigma)[n, m]$ and the family of term graphs with n variables and m roots. We introduce now a structure, called *computad* [30], that plays for 2-categories the same role that a graph with pairing plays for a gs-monoidal categories. Computads will be used in the next section to represent faithfully term graph rewriting systems (by providing a finitary encoding of the rules of a system). As shown below, a 2-category can be generated from a computad via a free construction.

Definition 16 (computads). A *computad* $\mathbf{C}_c$ is a pair $\langle \mathbf{C}, S \rangle$, where $\mathbf{C}$ is a category and S is a set of cells, each of which has two parallel arrows of $\mathbf{C}$ as *source* and *target*, respectively. A *c-morphism* $\langle F, h \rangle : \mathbf{C}_c \to \mathbf{C}'_c$ is a pair such that $F : \mathbf{C} \to \mathbf{C}'$ is a functor and $h : S \to S'$ is a function preserving source and target, that is, for every cell $\alpha : f \Rightarrow g \in S$ we have $h(\alpha) : F(f) \Rightarrow F(g) \in S'$.

A computad $\langle \mathbf{C}, S \rangle$ is gs-monoidal if so is $\mathbf{C}$, while a c-morphism $\langle F, h \rangle$ is gs-monoidal if so is F. Computads and c-morphisms form a category denoted **Comp**; also gs-monoidal computads and gs-monoidal c-morphisms form a category, denoted **GSM-Comp**. □

Proposition 17 (free gs-monoidal 2-categories). *Let V_2 :* **GSM-2Cat** $\to$ **GSM-Comp** *be the forgetful functor mapping a gs-monoidal 2-category to the underlying gs-monoidal computad (simply forgetting cell composition): It admits a left (free) adjoint* GSM_2 : **GSM-Comp** $\to$ **GSM-2Cat**. □

Intuitively, the free adjoint GSM_2 composes the cells of a computad in all the possible ways, both horizontally and vertically, imposing further equalities in order to satisfy the axioms of a 2-category and preserving the gs-monoidal structure on the underlying category.

4 Rewriting Sequences as Cells of a 2-Theory

This section presents the main result of the paper. We first show how to generate in a free way a gs-monoidal 2-category from a given term graph rewriting system

$\mathcal{R}$; next we prove that cells of this category faithfully correspond to plain rewriting sequences using the rules of $\mathcal{R}$. Part of the construction of the mentioned gs-monoidal 2-category is based on the main result presented in [8], namely that there is an isomorphism between the category of term graphs over Σ (Definition 5) and the gs-monoidal theory of Σ (Proposition 15).

Theorem 18 (the category of term graphs as gs-monoidal theory). *Let $\mathbf{TG}_\Sigma$ be the category introduced in Definition 5. Then the structure $\langle \mathbf{TG}_\Sigma, \oplus, \underline{0}, G_{\Pi(\_,\_)}, G_{\nabla(\_)}, G_{!}^{-} \rangle$ is a gs-monoidal category isomorphic to $\mathbf{GS\text{-}Th}(\Sigma)$, where:*

- *Functor $\oplus : \mathbf{TG}_\Sigma \times \mathbf{TG}_\Sigma \to \mathbf{TG}_\Sigma$ is defined on arrows as in Definition 6, and on objects as $\underline{n} \oplus \underline{m} = \underline{n+m}$;*
- *For each $n, m \in \mathbb{N}$, $G_{\Pi(\underline{n},\underline{m})}$ is the discrete term graph of rank $(n+m, n+m)$ such that (denoting by r and v the root and variable functions of one of its representatives) $r(x) = v(n+x)$ if $x \leq m$, and $r(x) = v(x-m)$ if $m < x \leq n+m$.*
- *For each $n \in \mathbb{N}$, $G_{\nabla(\underline{n})}$ is the discrete term graph of rank $(2*n, n)$ such that $r(x) = x$ if $x \leq n$, and $r(x) = x - n$ if $n < x \leq 2*n$.*
- *$G_{!}^{\underline{n}}$ is the discrete term graph with n variables and with no roots.* □

The gs-monoidal 2-category generated by a term graph rewriting system over a signature Σ will have $\mathbf{GS\text{-}Th}(\Sigma)$ as underlying category. The result just presented allows us to use term graphs over Σ to denote the arrows of that category, and to make free use of the equations of gs-monoidal categories when reasoning about them. We first show how to associate (in a quite straightforward way) with each rewriting system $\mathcal{R}$ its *gs-monoidal 2-theory* $\mathbf{GS\text{-}2Th}(\mathcal{R})$, and then relate cells in this 2-category and rewriting sequences over $\mathcal{R}$.

Definition 19 (rule as cell). Let $\mathbf{R} = \langle d, r_l, r_r \rangle$ be a rule, as in Definition 7. Let n be the number of empty nodes of d, and let m be the number of nodes of $d|r_l$. Then we define the term graphs $L_\mathbf{R}$ and $R_\mathbf{R}$ of rank (m, n) as follows:

- $L_\mathbf{R} = [\langle r, d|r_l, v \rangle]$, where $r : \underline{m} \to N(d|r_l)$ is a bijection, and $v : \underline{n} \to N_\emptyset(d|r_l)$ is a bijection between $\underline{n}$ and the set of empty nodes of $d|r_l$. Note that r and v are well-defined by the definition of n and m; moreover they could be fixed in a canonical way, but we don't need this.
- $R_\mathbf{R} = [\langle r', d - \{r_l\}, v \rangle]$, where $d - \{r_l\}$ is the dag obtained by removing from d the node r_l, v is the same as in the previous point, and for all $x \in \underline{m}$,

$$r'(x) = \begin{cases} r(x) & \text{if } r(x) \neq r_l \\ r_r & \text{if } r(x) = r_l \end{cases}$$

Again it is easy to check that $R_\mathbf{R}$ is a well-defined term graph of rank (m, n).

The cell representing rule $\mathbf{R}$ is the cell $\alpha(\mathbf{R}) : L_\mathbf{R} \Rightarrow R_\mathbf{R} : \underline{m} \to \underline{n}$. □

Definition 20 (the gs-monoidal 2-theory of a rewriting system). Let $\mathcal{R} = \langle \{\mathbf{R}_i\}_{i\in I}, \Sigma \rangle$ be a rewriting system, and let $\mathbf{GS\text{-}Th}(\Sigma)$ be the gs-monoidal theory of Σ. The *gs-monoidal computad representing* $\mathcal{R}$ is defined as the computad $\mathbf{C}_\mathcal{R} = \langle \mathbf{GS\text{-}Th}(\Sigma), \{\alpha(\mathbf{R}_i)\}_{i\in I}\rangle$, where for each rule $\mathbf{R}_i$ in $\mathcal{R}$ the cell $\alpha(\mathbf{R}_i) : L_{\mathbf{R}_i} \Rightarrow R_{\mathbf{R}_i}$ is as in Definition 19. The *gs-monoidal 2-theory of* $\mathcal{R}$, denoted by $\mathbf{GS\text{-}2Th}(\mathcal{R})$ is the free gs-monoidal 2-category generated by $\mathbf{C}_\mathcal{R}$, as described in Proposition 17. □

We need now a few technical definitions and lemmas before presenting the main result relating cells of $\mathbf{GS\text{-}2Th}(\mathcal{R})$ and rewriting sequences over $\mathcal{R}$.

Definition 21 (term graph contexts). A *(term graph) context* $C[i,j]$ is a well-formed, ranked term graph expression containing exactly one (ranked) placeholder $[i,j]$ (for some $i,j \in \mathbb{N}$). If $C[i,j]$ is a context and G is a term graph of rank (i,j), then by $C[G]$ we denote the term graph obtained by evaluating the expression C after replacing $[i,j]$ by G. □

Next lemma provides a “syntactical” characterization of the existence of a plain morphism between two term graphs by using contexts. Due to space limitations, the (quite long) proof will only appear in the full version of the paper.

Lemma 22 (plain morphisms and contexts). *Let G and H be two term graphs such that all nodes of G are roots. Then there is a context $C[m,n]$ (where (m,n) is the rank of G) such that $H = C[G]$ if and only if for every dag $d_G \in G$ and $d_H \in H$ there is a plain dag morphism $f : d_G \to d_H$ (see Definition 2).* □

The last lemma allows us to reformulate the definition of plain term graph rewriting in a more “algebraic” way.

Lemma 23 (term graph rewriting as contextualization). *Let $\mathbf{R} = \langle d, r_l, r_r\rangle$ be a rule and $L_\mathbf{R}, R_\mathbf{R}$ be the (m,n)-ranked term graphs of Definition 19. Then for every context $C[m,n]$ and for every dag d_0 in $C[L_\mathbf{R}]$ there is a plain redex $\Delta = \langle \mathbf{R}, f : d|r_l \to d_0\rangle$ such that $C[L_\mathbf{R}] \to_\Delta C[R_\mathbf{R}]$.*

Proof. Let $f : d|r_l \to d_0$ be the plain dag morphism whose existence is ensured by Lemma 22. The fact that $C[L_\mathbf{R}] \to_{\langle \mathbf{R}, f\rangle} C[R_\mathbf{R}]$ can be shown by comparing the construction in Definition 8 with the definition of $L_\mathbf{R}$ and $R_\mathbf{R}$. In fact the application of rule $\mathbf{R}$ to d_0 has the effect of adding to d_0 a copy of $d|r_r - d|r_l$ and of removing node $f(r_l)$, which is exactly the difference between $R_\mathbf{R}$ and $L_\mathbf{R}$. Furthermore, all references to $f(r_l)$ are redirected to $\hat{f}(r_r)$. The same happens by replacing $L_\mathbf{R}$ by $R_\mathbf{R}$ in an arbitrary context, because (by the conditions on rules in Definition 7) the only reference to r_l in $L_\mathbf{R}$ is a root, which is “redirected” to r_r in $R_\mathbf{R}$, by Definition 19. □

We are now ready to present our main result.

Theorem 24 (rewriting sequences as cells of the gs-monoidal 2-theory). *Let G and H be two term graphs having the same rank (m,n). Then there is a cell $\alpha : G \Rightarrow H : \underline{n} \to \underline{m}$ in $\mathbf{GS\text{-}2Th}(\mathcal{R})$ if and only if there is a plain rewriting sequence $G \to_\mathcal{R}^* H$.*

Proof. Only if part. Assume that there is a cell $\alpha : G \Rightarrow H : \underline{n} \to \underline{m}$ in $\mathbf{GS\text{-}2Th}(\mathcal{R})$. We prove the existence of a plain rewriting sequence from G to H by induction on the number of *2-generators* in α (i.e., the components of α of the form $\alpha(\mathbf{R})$, for $\mathbf{R}$ a rule in $\mathcal{R}$). Note that this is correct, because the axioms for gs-monoidal 2-categories do not identify cells containing a different number of 2-generators. If α does not contain any 2-generator, then it is an identity cell, and the statement trivially holds true. Suppose now that α contains $k+1$ 2-generators. Using essentially the *interchange law*, it is possible to decompose it into an equivalent cell $\alpha_1 \cdot \alpha_2 \cdot \ldots \cdot \alpha_{k+1}$ such that each α_i contains exactly one 2-generator. Let K be the target graph of α_1, i.e., $\alpha_1 : G \Rightarrow K$. Since α_1 contains one 2-generator, it must be of the form $C[\alpha(\mathbf{R})]$ for a suitable context C and a rule $\mathbf{R} \in \mathcal{R}$. By the definition of $\alpha(\mathbf{R})$, we have that $G = C[L_{\mathbf{R}}]$ and $K = C[R_{\mathbf{R}}]$, and by Lemma 23 there is a plain redex Δ such that $G \to_{\Delta} K$. The statement follows because by induction hypothesis over cell $\alpha_2 \cdot \ldots \cdot \alpha_{k+1} : K \Rightarrow H$ (which contains k 2-generators), there is a plain rewriting sequence from K to H.
If Part. If $G \to_{\Delta} H$ using a plain redex $\langle \mathbf{R}, f \rangle$ with $\mathbf{R} \in \mathcal{R}$, by Lemma 22 there is a context C_f such that $G = C_f[L_{\mathbf{R}}]$. Moreover, we have that $H = C_f[R_{\mathbf{R}}]$ by Lemma 23. By Definitions 19 and 20 the cell $\alpha(\mathbf{R}) : L_{\mathbf{R}} \Rightarrow R_{\mathbf{R}}$ is a 2-generator of $\mathbf{GS\text{-}2Th}(\mathcal{R})$, and thus (by the generation rules for cells) $C_f[\alpha(\mathbf{R})] : C_f[L_{\mathbf{R}}] \Rightarrow C_f[R_{\mathbf{R}}]$ is a cell. Then the statement follows because concatenation of rewriting steps corresponds faithfully to vertical composition of cells. □

5 Relationship with Term Rewriting

Our 2-categorical presentation of term graph rewriting is directly inspired by that of term rewriting introduced for example in [26, 25, 23]: the goal of this section is to relate the two theories by exploiting the categorical framework.

Firstly, let us briefly recall the relationship between the gs-monoidal theory of a signature Σ, $\mathbf{GS\text{-}Th}(\Sigma)$, and its algebraic theory, $\mathbf{Th}(\Sigma)$, whose definition dates back to Lawvere's thesis [21] (for a more detailed discussion about the various definitions of algebraic theories and the relationship with gs-monoidal theories we refer to [8]).

Definition 25 (algebraic theory over a signature). An *algebraic theory* $\mathbf{C}$ is a category whose objects are underlined natural numbers, and which for each $\underline{n}$ is equipped with an n-tuple of maps $\langle \pi_i^n : \underline{n} \to \underline{1} \mid i = 1 \ldots n \rangle$, making $\underline{n}$ the n-fold cartesian product of $\underline{1}$, that is, $\underline{n} = \underline{1}^n$.

Given a signature Σ, the *algebraic theory of* Σ is the algebraic theory $\mathbf{Th}(\Sigma)$ such that the hom-set $\mathbf{Th}(\Sigma)[\underline{n}, \underline{m}]$ is the set of m-tuples of terms over variables $\{x_1, \ldots, x_n\}$; for each $\underline{n} \in \mathbb{N}$, the required n-tuple of maps is given by $\langle x_1, \ldots, x_n \rangle$; and arrow composition is defined as term substitution. □

It can be shown that $\mathbf{Th}(\Sigma)$ is indeed an algebraic theory. An equivalent way of introducing the algebraic theory of Σ is via a free construction, as for gs-monoidal theories in Proposition 15. In fact, there is an adjunction between the category $\mathbf{M\text{-}Gr}$ and $\mathbf{C\text{-}Cat}$, a suitable category of cartesian categories, which factorizes through the adjunction mentioned in Proposition 15. This fact allows us to relate the gs-monoidal and the algebraic theories of a signature.

Proposition 26 (relating algebraic and gs-monoidal theories). *Let* **C-Cat** *be the category of small cartesian categories with strictly associative products as objects, and functors preserving products "on the nose" as arrows, and let* **GSM-Cat** *and* **M-Gr** *be the categories of Definition 13 and 14, respectively.*

$$
\begin{tikzcd}
& \mathbf{M\text{-}Gr} \arrow[dl, bend right, "GSM"'] \arrow[dr, bend left, "C_C"] & \\
\mathbf{GSM\text{-}Cat} \arrow[ur, "V"] \arrow[rr, "C"] & & \mathbf{C\text{-}Cat} \arrow[ul, "V_C"'] \arrow[ll, bend left, "U"]
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathbf{GS\text{-}Comp} \arrow[r, bend left, "C_{Comp}"] \arrow[r, phantom, "\perp"] \arrow[d, bend left, "GSM_2"] \arrow[d, phantom, "\vdash"] & \mathbf{C\text{-}Comp} \arrow[l, bend left] \arrow[d, bend left, "C_2"] \arrow[d, phantom, "\vdash"] \\
\mathbf{GS\text{-}2Cat} \arrow[r, bend left, "C^2"] \arrow[r, phantom, "\perp"] \arrow[u, bend left] & \mathbf{C\text{-}2Cat} \arrow[l, bend left, "U^2"] \arrow[u, bend left]
\end{tikzcd}
$$

Fig. 3. Two diagrams of adjunctions.

Then the left diagram of Figure 3 commutes, where V and V_C are the obvious forgetful functors that map a gs-monoidal or cartesian[6] *category to the underlying graph with pairing, C_C generates from a graph with pairing a cartesian category, by adding all the missing structure, U is an inclusion (because every cartesian category is gs-monoidal), GSM is as in Proposition 15, and C maps a gs-monoidal category M to the quotient category obtained by imposing the following naturality axioms for all objects a, b and arrows $f : a \to b \in M$:*

$$f; \nabla_b = \nabla_a; (f \otimes f) \qquad f; !_b = !_a$$

Furthermore, we have $\mathbf{Th}(\Sigma) = C_C(G_\Sigma)$, where G_Σ is as in Definition 14, and therefore $\mathbf{Th}(\Sigma) = C(\mathbf{GS\text{-}Th}(\Sigma))$.

Proof outline. To show that a cartesian category is gs-monoidal, define $\nabla_a = \langle id_a, id_a \rangle$ (using pairing), and $!_a$ as the arrow to the terminal object; then it is easy to see that all axioms are satisfied. Concerning functor C, the fact that $C(M)$ is cartesian is based on a result which can be traced back to the early Seventies [6], rediscovered and become folklore in recent years [15, 19]. The other functors in the diagram can be defined by a tedious but straightforward inspection of the object structure of the concerned categories. For a characterization of the algebraic theory as a free construction see for example [12]. Finally, the relationship between algebraic and gs-monoidal theories follows by uniqueness of left adjoints. □

[6] By cartesian we always mean "cartesian with strictly associative products".

The adjunction $C \dashv U : \mathbf{GSM\text{-}Cat} \to \mathbf{C\text{-}Cat}$ allows us to relate terms and term graphs over the same signature in a clean way. In fact, the component of the unit of the above adjunction on a gs-monoidal theory, i.e, $\eta_{\mathbf{GS\text{-}Th}(\Sigma)} : \mathbf{GS\text{-}Th}(\Sigma) \to U(C(\mathbf{GS\text{-}Th}(\Sigma))) = \mathbf{Th}(\Sigma)$, maps each term graph of rank (m, n) to the m-tuple of terms obtained by *unraveling* the term graph from the m roots. Such a mapping can be regarded as a clean categorical version of the unraveling function defined in many papers on term graph rewriting to extract a term from a term graph. Also, it provides a clear interpretation of the relationship between term and term graphs: *a term is an equivalence class of term graphs, modulo the naturality of* ∇ *and* $!$.

Let us consider now the relationship between the corresponding 2-theories. Let **C-Comp** be the category of computads having objects of **C-Cat** as underlying categories, and **C-2Cat** be the category of small *cartesian 2-categories*.[7] Then the right diagram of Figure 3 commutes, where the vertical adjunctions are as in Proposition 17 and the horizontal ones are based on the adjunction between **GSM-Cat** and **C-Cat** presented above.

These adjunctions allow us to relate term graph rewriting with term rewriting. Given a term graph rewriting system $\mathcal{R}$ over Σ and its computad $\mathbf{C}_{\mathcal{R}}$ (as in Definition 20), the cartesian computad $C_{comp}(\mathbf{C}_{\mathcal{R}})$ is a faithful representation of the term rewriting system $\mathcal{T}_{\mathcal{R}}$ obtained by transforming each rule $\langle d, r_l, r_r \rangle$ of $\mathcal{R}$ into the *term* rule $\eta(d|r_l) \to \eta(d|r_r)$, where $\eta(d')$ is, informally, the term obtained by unraveling dag d' from its root.

By uniqueness of left adjoints, we have $C^2(\mathbf{GS\text{-}2Th}(\mathcal{R})) = C^2(GSM_2(\mathbf{C}_{\mathcal{R}})) = C_2(C_{comp}(\mathbf{C}_{\mathcal{R}})) = \mathbf{2Th}(\mathcal{T}_{\mathcal{R}})$, where the last one is the algebraic 2-theory of system $\mathcal{T}_{\mathcal{R}}$, whose cells faithfully represent term rewriting sequences (see [25, 23, 9]). Therefore the unit of the adjunction $C^2 \dashv U^2$ maps each term graph rewriting sequence over $\mathcal{R}$ to a term rewriting sequence over $\mathcal{T}_{\mathcal{R}}$. Simple considerations about the 2-naturality of ∇ and $!$ show that this mapping removes all garbage rewritings, and duplicates a rewriting steps if the root of its redex is shared in the starting term graph.

6 Conclusion and Future Work

The main result of this paper shows that plain term graph rewriting sequences (without garbage collection) over a term graph rewriting system $\mathcal{R}$ have a faithful representation as cells of the gs-monoidal 2-theory of $\mathcal{R}$, i.e., a symmetric monoidal 2-category with some additional structure freely generated by the rules of $\mathcal{R}$. We also showed that the categorical framework relates in a clean way term and term graph rewriting. Space limitations prevent us from dealing with subjects like including garbage collection in this framework, weakening the restriction to plain rewriting sequences, and comparing our formalism with other notion of term graph rewriting. While for the first point all our constructions lift

[7] That is, 2-categories with strictly associative *2-products* and a *2-terminal* object, which are defined as the usual products and terminal objects, but where the universality statements are quantified over cells rather than over arrows.

smoothly (basically, we only need to assume the 2-naturality of the discharger !) making the comparison with other formalisms rather straightforward, weakening plain redexes is a key point, linked e.g. to classical problems of un/sharing in functional languages: Any other consideration is deferred to the full paper.

The representation of rewriting sequences as cells induces an obvious equivalence relation on them, relating two sequences if they are represented by the same cell. In the case of term rewriting, such equivalence coincides with the so-called *permutation equivalence* [5, 14], due to the axioms of cartesian 2-categories [20, 11]. The precise characterization of this equivalence for the case of term graph rewriting is left as a topic for further research. A promising direction consists in relating the definition of term graph rewriting presented here with the algebraic approach to graph transformation for which a rich theory concerning aspects of concurrency and parallelism has been developed [7], including various notion of equivalence on graph transformation sequences.

Another interesting topic is the generalization of the results presented in this paper and in [8] to the case of possibly cyclic term graphs, and to relate them to rational terms, i.e., possibly infinite terms having a finite number of distinct subterms. Such a topic is already considered for example in [17], but it would be interesting to see if it could be addressed using categorical techniques along the line of this paper. Some preliminary considerations suggest that *Iteration Theories* [4] could fruitfully be used to this aim, in place of algebraic theories.

References

1. Z.M. Ariola and J.W. Klop. Equational Term Graph Rewriting. *Fundamenta Informaticae*, 26:207–240, 1996.
2. H.P. Barendregt, M.C.J.D. van Eekelen, J.R.W. Glauert, J.R. Kennaway, M.J. Plasmeijer, and M.R. Sleep. Term graph reduction. In *Proceedings PARLE*, volume 259 of *LNCS*, pages 141–158. Springer Verlag, 1987.
3. M. Barr and C. Wells. *Category Theory for Computing Science.* Prentice Hall, 1990.
4. S. Bloom and Z. Ésik. *Iteration Theories.* EATCS Monographs on Theoretical Computer Science. Springer Verlag, 1991.
5. G. Boudol. Computational semantics of term rewriting systems. In M. Nivat and J. Reynolds, editors, *Algebraic Methods in Semantics*, pages 170–235. Cambridge University Press, 1985.
6. L. Budach and H.-J. Hoenke. *Automaten und Functoren.* Akademie-Verlag, 1975.
7. A. Corradini. Concurrent Graph and Term Graph Rewriting. In U. Montanari and V. Sassone, editors, *Proceedings CONCUR'96*, volume 1119 of *LNCS*, pages 438–464. Springer Verlag, 1996.
8. A. Corradini and F. Gadducci. An algebraic presentation of term graphs, via ps-monoidal categories. Submitted for publication. Available at `http://www.di.unipi.it/~gadducci/papers/aptg.ps`, 1997.
9. A. Corradini, F. Gadducci, and U. Montanari. Relating two categorical models of term rewriting. In *Rewriting Tecniques and Applications*, volume 914 of *LNCS*, pages 225–240. Springer Verlag, 1995.

10. A. Corradini and F. Rossi. Hyperedge Replacement Jungle Rewriting for Term Rewriting Systems and Logic Programming. *Theoret. Comput. Sci.*, 109:7–48, 1993.
11. G. Gadducci. *On the Algebraic Approach to Concurrent Term Rewriting*. PhD thesis, University of Pisa - Department of Computer Science, 1996.
12. J.A. Goguen, J.W. Tatcher, E.G. Wagner, and J.R Wright. Some Fundamentals of Order-Algebraic Semantics. In *Mathematical Foundations of Computer Science*, volume 45 of *LNCS*, pages 153–168. Springer Verlag, 1976.
13. B. Hoffmann and D. Plump. Implementing Term Rewriting by Jungle Evaluation. *Informatique théorique et Applications/Theoretical Informatics and Applications*, 25:445–472, 1991.
14. G. Huet and J-J. Lévy. Computations in Orthogonal Rewriting Systems, I. In J.-L. Lassez and G. Plotkin, editors, *Computational Logic: Essays in honour of J. A. Robinson*, pages 395–414. MIT Press, 1991.
15. B. Jacobs. Semantics of Weakening and Contraction. *Annals of Pure and Applied Logic*, 69:73–106, 1994.
16. G.M. Kelly and R.H. Street. Review of the elements of 2-categories. volume 420 of *Lecture Notes in Mathematics*, pages 75–103. Springer Verlag, 1974.
17. J.R. Kennaway, J.W. Klop, M.R. Sleep, and F.J. de Vries. On the Adequacy of Graph Rrewriting for Simulating Term Rewriting. *ACM Trans. Program. Lang. Syst.*, 16:493–523, 1994.
18. A. Kock and G.E. Reyes. Doctrines in Categorical Logic. In J. Bairwise, editor, *Handbook of Mathematical Logic*, pages 283–313. North Holland, 1977.
19. Y. Lafont. Equational Reasoning with 2-dimensional Diagrams. In *Term Rewriting, French Spring School of Theoretical Computer Science*, volume 909 of *LNCS*, pages 170–195. Springer Verlag, 1995.
20. C. Laneve and U. Montanari. Axiomatizing permutation equivalence in the λ-calculus. *Mathematical Structures in Computer Science*, 6:219–249, 1996.
21. F.W. Lawvere. Functorial Semantics of Algebraic Theories. *Proc. National Academy of Science*, 50:869–872, 1963.
22. S. Mac Lane. *Categories for the working mathematician*. Springer Verlag, 1971.
23. J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoret. Comput. Sci.*, 96:73–155, 1992.
24. M.J. Plasmeijer and M.C.J.D. van Eekelen. *Functional Programming and Parallel Graph Rewriting*. Addison Wesley, 1993.
25. A.J. Power. An abstract formulation for rewrite systems. In *Proceedings Category Theory in Computer Science*, volume 389 of *LNCS*, pages 300–312. Springer Verlag, 1989.
26. D.E. Rydehard and E.G. Stell. Foundations of equational deductions: A categorical treatment of equational proofs and unification algorithms. In *Proceedings Category Theory in Computer Science*, volume 283 of *LNCS*, pages 114–139. Springer Verlag, 1987.
27. M.R. Sleep, M.J. Plasmeijer, and M.C. van Eekelen, editors. *Term Graph Rewriting: Theory and Practice*. Wiley, London, 1993.
28. J. Staples. Computation of graph-like expressions. *Theoret. Comput. Sci.*, 10:171–195, 1980.
29. J.G. Stell. *Categorical Aspects of Unification and Rewriting*. PhD thesis, University of Manchester, 1992.
30. R. Street. Categorical structures. In M. Hazewinkel, editor, *Handbook of Algebra*, pages 529–577. Elsevier, 1996.