On GS-Monoidal Theories for Graphs with Nesting

Roberto Bruni[1], Andrea Corradini[1], Fabio Gadducci[1], Alberto Lluch Lafuente[2], and Ugo Montanari[1]

[1] Dipartimento di Informatica, Università di Pisa, Italy
{bruni,andrea,gadducci,ugo}@di.unipi.it
[2] IMT Institute for Advanced Studies, Lucca, Italy
alberto.lluch@imtlucca.it

Abstract. We propose a sound and complete axiomatisation of a class of graphs with nesting and either locally or globally restricted nodes. Such graphs allow to represent explicitly and at the right level of abstraction some relevant topological and logical features of models and systems, including nesting, hierarchies, sharing of resources, and pointers or links. We also provide an encoding of the proposed algebra into terms of a gs-monoidal theory, and through these into a suitable class of "well-scoped" term graphs, showing that this encoding is sound and complete with respect to the axioms of the algebra.

1 Introduction

The use of graphs or diagrams of various kinds is pervasive in Computer Science, as they are very handy for describing in a two-dimensional space the logical or topological structure of systems, models, states, behaviours, computations, and several other entities of interest; the reader might be familiar, for example, with the graphical presentations of entity-relationship diagrams, of finite state automata, of static and behavioural UML diagrams (like class, message sequence and state diagrams), of computational formalisms like Petri nets, and so on.

The advantage of using graphs or diagrams, rather than a linear syntax based on terms or strings, lies in the fact that graphs can represent in a direct way relevant topological features of the systems/models they describe, including nesting, hierarchies, sharing of structures, and pointers or links, among others, making such features easily understandable also to non-specialists. In several cases graphs provide a representation of models or systems at the "right" level of abstraction: often a single graph corresponds to an equivalence class of terms, up to an axiomatic specification equating systems considered as topologically indistinguishable. Furthermore, as drawings are always understood "up to isomorphism", if the concrete identities of certain syntactical entities are irrelevant (for example, the name of the states of a finite state automata), it is sufficient to avoid depicting them in the drawing (a state is uniquely identified by the graphical components it is represented with).

Another interesting case where a graphical syntax allows one to get rid of irrelevant information from the linear syntax is the representation of terms or formulæ of languages with binding operators (like first-order formulæ with quantifiers, λ-terms with abstractions, terms of process calculi with name restriction operators, among others). In all such cases, the classical linear syntax actually considers terms up to α-conversion (i.e., non-capturing renaming of bound variables or names): a graphical syntax can easily handle this, by representing a bound variable/name with an unlabelled node and references to it with edges.

Unfortunately, though, graphical representations are much more difficult to handle and to analyse than linear ones. In general, a graphical model needs to be encoded into a linear syntax, in order to exploit tools like theorem provers or model checkers to verify certain properties on it. Typically, a linear syntax can be defined by introducing an equational signature, whose operator symbols are interpreted as operations on graphs and where the axioms formalise suitable properties of these operators: then the terms of the initial algebra can be interpreted as graphs.

In this paper we are concerned with graphical notations that treat as first-class citizens the sharing of possibly bound names as sketched above, as well as the nesting of structures, a feature emerging as a recurrent pattern in the conceptual modelling of systems: on the informatics side, one may think of file systems, composite diagrams, networks, sessions, transactions, locations, structured state machines, or even XML documents, among others; or one may consider natural models of computations, like those arising in bioinformatics, equating nesting with the presence of membranes for molecules in chemical compounds.

As a main contribution, we introduce in Section 2 a visual modelling framework suited for representing systems exhibiting nesting of structures and sharing of atomic, named resources, as well as both local and global restriction. This framework consists not only of a class of hierarchical graphs, called *NR-graphs*, which allow to represent such systems in a direct, intuitive way, but also of a standard algebraic presentation, made of a signature and a set of axioms, defining the *algebra of graphs with nesting*, called **AGN**. The two components are related by a formal result stating that the proposed axiomatisation is sound and complete, i.e., that equivalence classes of terms of the algebra are in bijective correspondence with NR-graphs taken up to isomorphism: this relationship is represented by the top horizontal arrow of the next diagram.

$$\begin{tikzcd}
\mathbf{AGN}(S,B)/_{\equiv_{\mathcal{A}}} \arrow[r, Leftrightarrow] \arrow[d, hook, "Sec.\ 4"'] & \text{NR-Graphs over } (S,B) \arrow[d, hook, "Sec.\ 5"] \\
\mathbf{GS}(\Sigma^{\bullet}_{B}) \arrow[r, leftrightarrow, "{[10]}"] & \text{Term Graphs over } \Sigma^{\bullet}_{B}
\end{tikzcd}$$

This result is not proved directly, but it is obtained, indirectly, by relating the newly introduced framework of NR-graphs to the well-developed theory of *term graphs*. Roughly, term graphs, presented in Section 3, are directed acyclic graphs over a signature Σ, and they intuitively represent "terms with shared sub-terms" over Σ: therefore they are inherently simpler than our NR-graphs (they feature neither nesting nor restriction). An algebraic, equational characterisation of term graphs was proposed in [10] exploiting the so-called *gs-monoidal theories*: it is analogous to the characterisation of terms built over a signature Σ as arrows of a cartesian category, the Lawvere theory of Σ, freely generated from the signature. The bijective correspondence between term graphs and equivalence classes of gs-monoidal terms is represented by the bottom arrow of the above diagram.

To relate the two formalisms, we first present in Section 4 an encoding of the terms of the algebra **AGN** into terms of the gs-monoidal theory (the left vertical arrow above): this translation is shown to be sound and complete, i.e., two terms are mapped to equivalent gs-monoidal terms if and only if they are equivalent. Intuitively, this is possible because the nesting of nodes and edges is encoded faithfully by exploiting a new sort of the signature of term graphs, introduced to represent *locations*; furthermore, global and local restriction are represented with suitable operators. Next in Section 5 we show that there is a bijective correspondence between NR-graphs and "well-scoped" term graphs (the right vertical arrow above): through the composition of this bijection with the encoding of **AGN** terms as gs-monoidal terms we obtain the bijection between the terms of the algebra and NR-graphs.

The bridge between the theories of graphs with nesting and of term graphs established by the proposed encoding can be used to exploit in the newly introduced framework the rich theory and pragmatics developed for term graphs and term graph rewriting along the years: we sketch some possible developments in Section 6. We conclude by reviewing some relevant related works in Section 7 and by proposing some topics of future research in Section 8.

2 An Algebra for Graphs with Nesting and Restrictions

Many notions of hierarchically structured graphs were proposed in the literature (see Section 7), mostly defined set-theoretically as (hyper-)graphs whose nodes and edges can be related to other graphs (e.g., by suitable containment morphisms or adjacency relations). Rarely such graphs come equipped with a handy syntax for representing and manipulating them algebraically, a topic on which we recently made some progress, building on previously developed notations like CHARM [11] and other syntactic formalisms for representing plain graphs with interfaces [14]. We have been also influenced by the notation used in nominal calculi.

In the present section we introduce the *graphs with nesting and restriction*, briefly *NR-graphs*, as well as an equational axiomatisation for them: the main result states that equivalence classes of terms of the algebra modulo the axioms are in bijective correspondence with isomorphism classes of graphs. The hierarchical graphs we present are based on the following rationale: 1) for flexibility and visual expressiveness we prefer to deal with hyper-graphs instead of ordinary two-ended arcs; 2) nesting is seen according to a classical boxes-within-boxes scheme and applied to edges, while nodes are seen mainly as attaching points without further containment; 3) nodes can be localised within a single edge, in which case they are made private and not visible "outside"; 4) nodes can also be globally available to allow connections across the nesting hierarchy.

2.1 Graphs with Nesting and Restriction

Let us introduce some formal notation. Given a set M we denote by M^* the free monoid over M, i.e., the set of finite lists of elements drawn from M, often denoted by an over-lined letter. The unit of M^* is denoted by ϵ. Depending on the context, we can find it more convenient to denote concatenation just by juxtaposition (like in $\overline{e} = e_1e_2...e_n$) or with commas (like in $\overline{e} = e_1, e_2, ..., e_n$). Given a list $\overline{e}$ we denote by $\overline{e}|_i$ its i-th element and by $|\overline{e}|$ the underlying set of list elements. For an ordinal n, we write $\underline{n}$ for the set $\{1, \ldots, n\}$. We overload $\#\_$ to denote both the length of a list and the cardinality of a set. We use the symbol $\uplus$ for the disjoint union of sets. We write $s \in \overline{e}$ as a shorthand for $s \in |\overline{e}|$. If $f : A \to B$ is a function, we denote by f^* its obvious monoidal extension $f^* : A^* \to B^*$ and by f itself the power-set extension $f : 2^A \to 2^B$.

All along this section, let S be a set of sorts, B be a ranked set of *box labels* with $rnk(b) \in S^*$ for all $b \in B$, and $\mathcal{X}$ be a countable set of names. A *node* is a pair $x : s \in \mathcal{X} \times S$, and the operator $\tau : (\mathcal{X} \times S)^* \to S^*$ applied to a list of nodes returns the list of their sorts. Note that nodes with the same name but with different sorts are kept distinct, e.g., $x : s_1 \neq x : s_2$ if $s_1 \neq s_2$.

We introduce now the formal definition of our hierarchical graphs. We define the set **NR-Graph** of *graphs with nesting and restriction* and, for a set X of nodes, the set **NR-Graph**$[X]$ of such graphs *with external nodes* X.

Definition 1 (NR-graphs). *An* NR-graph $G \in$ **NR-Graph** *is a tuple* $G = \langle FN, GR, H\rangle$*, where* FN *is a set of* free nodes*,* GR *is a set of* globally restricted nodes *with* $FN \cap GR = \emptyset$*, and* $H \in$ **NR-Graph**$[FN \cup GR]$*. The set of* global nodes *of* G *is given by* $FN \cup GR$*.*

An NR-graph H with external nodes X*,* $H \in$ **NR-Graph**$[X]$*, is a tuple* $H = \langle LR, E, l, c, \rho\rangle$*, where* LR *is a set of* locally restricted nodes *(satisfying* $LR \cap X = \emptyset$*),* E *is a set of (hyper-)edges,* $l : E \to B$ *labels each edge with an element of* B*,* $c : E \to (X \cup LR)^*$ *is the* connection function *(satisfying* $\tau(c(e)) = rnk(l(e))$ *for all* $e \in E$*), and* $\rho : E \to$ **NR-Graph**$[X \cup LR]$ *maps each edge to a graph nested within it.*

The *depth* of an NR-graph with external nodes $H = \langle LR, E, l, c, \rho\rangle$ is 0 if E is empty, and $depth(H) = 1 + max_{e\in E}\{depth(\rho(e))\}$ otherwise. The depth of an NR-graph $G = \langle FN, GR, H\rangle$ is the depth of H. We will consider only graphs of finite depth and with finite sets of edges and nodes.

Figure 1 shows a sample NR-graph of depth 3 which represents a network system comprising different subnets (net-labelled edges) and workstations (st) connected through links according to different patterns: each subnet has a single access point, while each workstation is attached to two connection hubs (s-labelled nodes). The top level of the graph is defined as $G = \langle\{x : s\}, \{y : s\}, H_0\rangle$, with $H_0 = \langle\emptyset, \{e\}, \{e \mapsto \mathsf{net}\}, \{e \mapsto y\}, \{e \mapsto H_1\}\rangle$. G and H_0 define the global

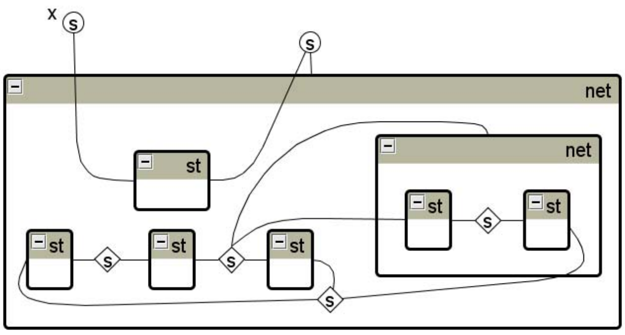


Fig. 1. A sample NR-graph, called G

nodes and the outer net-labelled edge, while H_1, which is going to be defined immediately below, represents the graph internal to edge e.

Note that the global nodes are depicted, with a round shape, at the top of the graph, as conceptually they do not pertain to any particular location. Also the name is depicted for free nodes (only $x : s$, in this case), but not for globally restricted nodes (like y). All nodes are sorted: the sort is indicated by the inside label, and in this example all nodes have the same sort s, therefore sometimes we will omit it. Here is part of the definition of graph H_1:

$$\langle\{z_1, z_2, z_3\}, \{f_1, f_2, f_3, f_4, f_5\}, \{f_1 \mapsto \mathsf{st}, \ldots\}, \{f_1 \mapsto x \cdot y, \ldots\}, \{f_1 \mapsto \emptyset_{NR}, \ldots\}\rangle$$

For the sake of brevity, we omit the rest of the definition of the NR-graph, but it should be clear from the drawing. H_1 has three locally restricted nodes, that are depicted as diamonds: they are private to the immediately enclosing edge e and cannot be referenced from the outside, but they can be shared by the subgraphs nested within the edge. H_1 also contains five edges: f_1 is the one labelled by st, connected to the two global nodes, and not containing anything, which is represented by $\rho(f_1) = \emptyset_{NR}$ ($\emptyset_{NR}$ is the empty NR-graph). Notice that edges are represented as ranked boxes, possibly nested, with a label in the upper-right corner. The rank information consists of the list of sorted tentacles attached to the box (where the sort of the tentacle is the sort of the node it is attached to). The ordering of the tentacles is left implicit by counting in clockwise order, starting from the leftmost tentacle. Nesting of edges and nodes within other edges is given by spatial containment.

Actually, Fig. 1 does not show precisely the above defined NR-graph G, but rather its isomorphism class, according to the definition that follows.

Definition 2 (NR-Graph isomorphism). *Let $G = \langle FN, GR, H\rangle$ and $G' = \langle FN', GR', H'\rangle$ be two NR-graphs. We say that G and G' are* isomorphic*, written $G \cong G'$, if $FN = FN'$, there is an isomorphism $\phi\colon FN \cup GR \to FN' \cup GR'$ such that $\phi(x) = x$ for all $x \in FN$, and H is ϕ-isomorphic to H', written $H \cong_\phi H'$.*

Let X and X' be two sets of nodes, and $\phi\colon X \to X'$ be an isomorphism. Furthermore, let $H = \langle LR, E, l, c, \rho\rangle$ be in **NR-Graph**$[X]$ *and $H' = \langle LR', E', l', c', \rho'\rangle$*

be in **NR-Graph**$[X']$. *Then* $H \cong_\phi H'$ *if there exist isomorphisms* $\phi_L \colon LR \to LR'$ *and* $\phi_E \colon E \to E'$ *such that, calling* $\hat{\phi} \colon X \cup LR \to X' \cup LR'$ *the isomorphism induced by* ϕ *and* ϕ_L*, for all* $e \in E$ *it holds*

- $l'(\phi_E(e)) = l(e)$,
- $c'(\phi_E(e)) = \hat{\phi}^*(c(e))$, *and*
- $\rho'(\phi_E(e)) \cong_{\hat{\phi}} \rho(e)$.

Thus isomorphisms preserve the identity of free nodes, but not the one of restricted nodes and edges: this explains why only the identities of free nodes are depicted in Fig. 1.

2.2 The Algebra for Graphs with Nesting

Even if the graphical representation of a system like the one of Fig. 1 is pretty intuitive and easy to understand for human beings, it might not be usable, e.g., as the input of a verification tool needed to analyse it. On the other hand, the set-theoretical presentation according to Definition 1 does provide a linear syntax for such graphs, but it is quite involved, as it emerges from the (partial) definition given by the graphs G, H and H_1 above. The main motivation of the graph algebra we are going to introduce is to provide a much more compact and intuitively understandable linear syntax for NR-graphs.

Definition 3 (algebra for graphs with nesting and restrictions, AGN). *The terms of the* algebra for graphs with nesting *(or* nested graphs*) are generated according to the following grammar*

$$\mathbb{G} ::= \mathbf{0} \mid x : s \mid b[\mathbb{G}](\overline{y}) \mid \mathbb{G} \mid \mathbb{G} \mid (\nu\, x : s)\mathbb{G} \mid (\mu\, x : s)\mathbb{G}$$

where $x : s$ *is a node,* $b \in B$*, and* $\overline{y} \in (\mathcal{X} \times S)^*$ *is a list of nodes such that* $rnk(b) = \tau(\overline{y})$.

Roughly, $\mathbf{0}$ denotes the empty graph; $x : s$ is a discrete graph with a single node named x of sort s; $b[\mathbb{G}](\overline{y})$ is a hyper-edge labelled b, whose tentacles are attached to nodes $\overline{y}$ and enclosing the graph $\mathbb{G}$; $\mathbb{G} \mid \mathbb{H}$ is the disjoint union of graphs $\mathbb{G}$ and $\mathbb{H}$ up to common (free) nodes; finally, $(\nu\ x : s)\mathbb{G}$ and $(\mu\ x : s)\mathbb{G}$ denote the graph $\mathbb{G}$ after making node $x : s$ not visible from the outside (borrowing nominal calculus jargon, we say that the node $x : s$ is *restricted*). An **AGN** term where no edge $b[\mathbb{G}](\overline{y})$ appears is called *discrete* and usually denoted by $\mathbb{D}$. Recall that each label $b \in B$ has a fixed rank $rnk(b) \in S^*$: we only allow *well-sorted* graphs, where for any sub-term $b[\mathbb{G}](\overline{y})$ we have that the (lists of) sorts of b and $\overline{y}$ coincide (as required by the constraint $rnk(b) = \tau(\overline{y})$ in Definition 3).

Notably, we distinguish two kinds of restrictions: $(\mu\ x : s)\mathbb{G}$ is called *localised restriction*, meaning that the node $x : s$ resides together with the topmost edges of $\mathbb{G}$, while $(\nu\ x : s)\mathbb{G}$ is called *global restriction*, meaning that the location of $x : s$ is immaterial. The key difference is that when a graph is enclosed within an edge, its globally restricted nodes can traverse up the hierarchy (see axiom A8

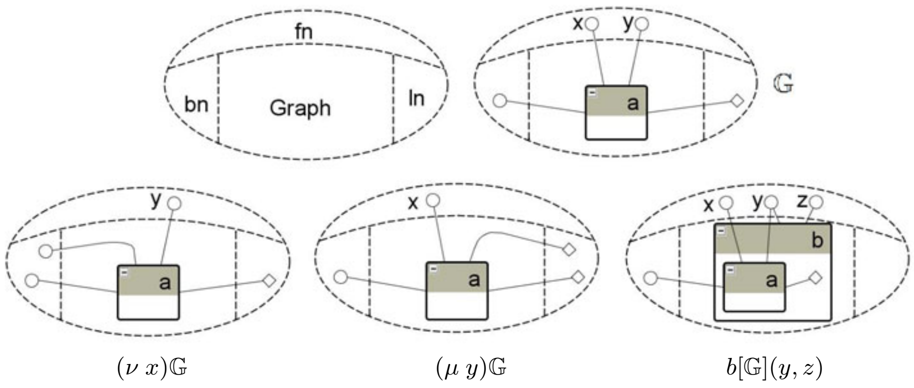


Fig. 2. Restrictions and nesting illustrated schematically

in Definition 5), while this is not the case for locally restricted nodes. When it is not necessary to distinguish which kind of restriction is considered, we write $(\omega\ x : s)$ using the wildcard $\omega \in \{\nu, \mu\}$. An **AGN** term where global restriction does not occur is called *ν-free.* Restrictions $(\nu\ x{:}s)\mathbb{G}$ and $(\mu\ x{:}s)\mathbb{G}$ act as binders for $x : s$ in $\mathbb{G}$, leading to the ordinary notion of *free nodes.*

Definition 4 (free nodes). *The set of* free nodes *of an* **AGN** *term* $\mathbb{G}$*, denoted* $fn(\mathbb{G})$*, is defined inductively as follows:*

$$fn(\mathbf{0}) \triangleq \emptyset \qquad fn(x : s) \triangleq \{x : s\} \qquad fn(b[\mathbb{G}](\overline{y})) \triangleq fn(\mathbb{G}) \cup |\overline{y}|$$

$$fn(\mathbb{G} \mid \mathbb{H}) \triangleq fn(\mathbb{G}) \cup fn(\mathbb{H}) \qquad fn((\omega\ x : s)\mathbb{G}) \triangleq fn(\mathbb{G}) \setminus \{x : s\}$$

As useful shorthands, we shall write $b(\overline{y})$ instead of $b[\mathbf{0}](\overline{y})$ and $b[\mathbb{G}]$ instead of $b[\mathbb{G}]()$: intuitively, the former denotes a plain edge, while the latter denotes a "floating" box (not anchored to any node). Moreover, we write $\prod_{i=1}^{n} \mathbb{G}_i$ as a shorthand for $\mathbb{G}_1 \mid (\mathbb{G}_1 \mid (\ldots \mid \mathbb{G}_n)\ldots))$ and we let $(\omega\ \overline{y})\mathbb{G}$ stand for the term $(\omega\ \overline{y}|_1)(\omega\ \overline{y}|_2)...(\omega\ \overline{y}|_m)\mathbb{G}$, where $m = \#\overline{y}$.

Figures 2 and 3 show the general idea for interpreting the operators of our algebra. We depict a graph as a large oval (see Fig. 2, top-left), with separated sectors for free nodes (top sector), globally (ν-) restricted nodes (left sector), top-level locally (μ-) restricted nodes (right sector) and all other nodes and edges (central sector). This is exemplified by a schematic graph drawn in Fig. 2, top-right, with a single edge attached to a few representative nodes (at least one of each kind, omitting their sorts): let us call it $\mathbb{G}$. The second line of Fig. 2 shows the graphs $(\nu\ x)\mathbb{G}$ (node x is moved from the sector of free nodes to that of globally restricted nodes), $(\mu\ y)\mathbb{G}$ (node y is moved from the sector of free nodes to that of locally restricted nodes) and $b[\mathbb{G}](y, z)$ (free nodes are shared between the enclosing edge and graph $\mathbb{G}$, globally bound nodes are preserved, localised nodes are enclosed in the top edge, leaving the right sector empty). Figure 3 shows the parallel composition of two generic graphs, obtained by taking the union of their free nodes and the disjoint union of all the other elements.

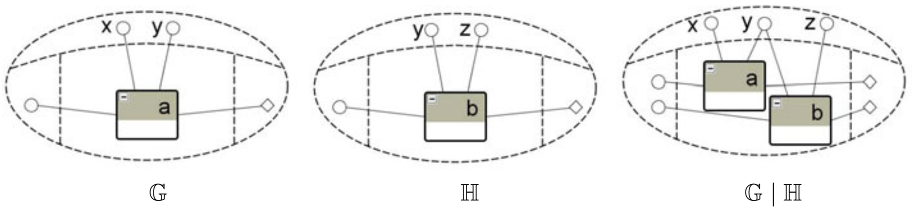


Fig. 3. Parallel composition illustrated schematically

Example 1. Figure 4 shows some graphs corresponding to simple terms of our algebra. Starting from top-left and in left-to-right reading direction we find the discrete graph $x : s$, the ordinary plain graphs $\mathsf{st}(x : s, y : s)$ and $\mathbb{G}_1 \triangleq \mathsf{st}(x, y) \mid \mathsf{st}(y, z)$, the graphs with restricted nodes $(\nu\ y)\mathbb{G}_1$ and $(\mu\ y)\mathbb{G}_1$, and the graphs with nesting $\mathsf{net}[(\nu\ y)\mathbb{G}_1](z)$ and $\mathsf{net}[(\mu\ y)\mathbb{G}_1](z)$.

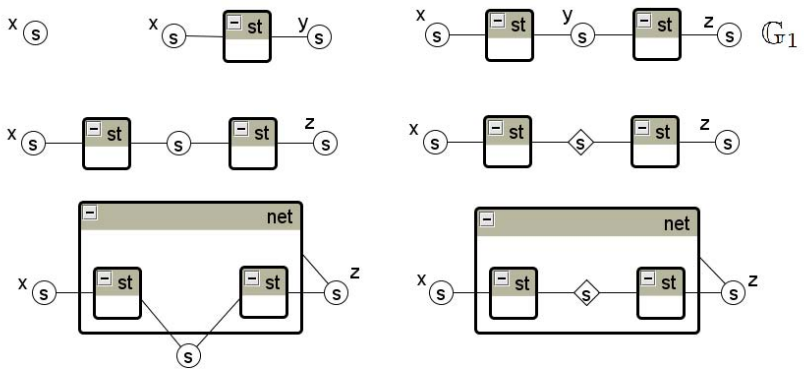


Fig. 4. Simple examples: $x{:}s$ (top-left), $\mathsf{st}(x{:}s, y{:}s)$ (top-centre), $\mathbb{G}_1 \triangleq \mathsf{st}(x, y) \mid \mathsf{st}(y, z)$ (top-right), $(\nu\ y)\mathbb{G}_1$ (mid-left), $(\mu\ y)\mathbb{G}_1$ (mid-right), $\mathsf{net}[(\nu\ y)\mathbb{G}_1](z)$ (bottom-left), $\mathsf{net}[(\mu\ y)\mathbb{G}_1](z)$ (bottom-right)

The terms of our algebra are too concrete, in the sense that different terms may intuitively correspond to the same nested graph (for example, the order in which we list the edges is obviously immaterial in the graph). Next we provide an axiomatisation equating those terms that define essentially the same graph.

The axiomatisation includes the structural graph axioms of [11] such as associativity and commutativity for $\mid$ with identity $\mathbf{0}$ (axioms A1–A3) and node restriction binding (A4–A6). It additionally includes axioms to α-rename bound nodes (A7), an axiom for the extrusion of globally bound, nested nodes (A8) that marks the distinction between global restriction ν and local restriction μ, an axiom for making immaterial the addition of a node to a graph where that same node is already free (A9) and an axiom ensuring that global nodes are not localised in lower layers (A10).

Definition 5 (structural congruence $\equiv_{\mathcal{A}}$). *The structural congruence $\equiv_{\mathcal{A}}$ over nested graphs is the least congruence satisfying*

$$\begin{array}{rcll}
\mathbb{G} \mid \mathbb{H} &\equiv& \mathbb{H} \mid \mathbb{G} & & \text{(A1)}\\
\mathbb{G} \mid (\mathbb{H} \mid \mathbb{I}) &\equiv& (\mathbb{G} \mid \mathbb{H}) \mid \mathbb{I} & & \text{(A2)}\\
\mathbb{G} \mid \mathbf{0} &\equiv& \mathbb{G} & & \text{(A3)}\\
(\omega_1\ x:s)(\omega_2\ y:t)\mathbb{G} &\equiv& (\omega_2\ y:t)(\omega_1\ x:s)\mathbb{G} & \textit{if}\ x:s \neq y:t & \text{(A4)}\\
(\omega\ x:s)\mathbf{0} &\equiv& (\omega\ x:s)x:s & & \text{(A5)}\\
\mathbb{G} \mid (\omega\ x:s)\mathbb{H} &\equiv& (\omega\ x:s)(\mathbb{G} \mid \mathbb{H}) & \textit{if}\ x:s \notin \mathit{fn}(\mathbb{G}) & \text{(A6)}\\
(\omega\ x:s)\mathbb{G} &\equiv& (\omega\ y:s)(\mathbb{G}\{^{y:s}/_{x:s}\}) & \textit{if}\ y:s \notin \mathit{fn}(\mathbb{G}) & \text{(A7)}\\
b[(\nu\ x:s)\mathbb{G}](\overline{y}) &\equiv& (\nu\ x:s)b[\mathbb{G}](\overline{y}) & \textit{if}\ x:s \notin |\overline{y}| & \text{(A8)}\\
x:s \mid \mathbb{G} &\equiv& \mathbb{G} & \textit{if}\ x:s \in \mathit{fn}(\mathbb{G}) & \text{(A9)}\\
b[x:s \mid \mathbb{G}](\overline{y}) &\equiv& x:s \mid b[\mathbb{G}](\overline{y}) & & \text{(A10)}
\end{array}$$

where $\{^{y:s}/_{x:s}\}$ denotes the capture-avoiding substitution of $x:s$ by $y:s$ and $\omega, \omega_1, \omega_2$ range over $\{\nu, \mu\}$.

It is immediate to observe that structural congruence respects free nodes, i.e., $\mathbb{G} \equiv_{\mathcal{A}} \mathbb{H}$ implies $\mathit{fn}(\mathbb{G}) = \mathit{fn}(\mathbb{H})$ for any $\mathbb{G}, \mathbb{H}$.

Next statement establishes the soundness and the completeness of the proposed axiomatisation: the proof is based on the results presented in later sections.

Theorem 1. *The equivalence classes of terms of algebra* **AGN** *modulo $\equiv_{\mathcal{A}}$ are in bijective correspondence with the isomorphism classes of NR-graphs.*

Proof. The statement will follow from Propositions 2, 3 and 4.

The translation of **AGN** terms into NR-graphs is sketched above in Figs. 2 and 3. Vice versa, the intuitive way to express a nested graph as an **AGN** term is to start writing the discrete term corresponding to the free nodes of the graph, then to add arbitrary distinct names for the top-level unnamed nodes, with the corresponding ν- and μ-restrictions, and finally to list all the top-level edges, properly attached to the available nodes, and with this procedure applied inductively to the contents of each edge.

To conclude this section, let us show how the proposed algebra meets the goal of providing a concise and intuitive linear syntax for NR-graphs.

Example 2. The graph G in Fig. 1 corresponds to the following term $\mathbb{G}_G$ (where we omit the node sorts, all equal to s)

$$\begin{array}{l}(\nu\ y)\mathsf{net}[\ \mathsf{st}(x,y)\ \mid\ (\mu\ z_1,z_2,z_3)(\ \mathsf{st}(z_3,z_1)\ \mid\ \mathsf{st}(z_1,z_2)\ \mid\ \mathsf{st}(z_2,z_3)\ \mid\\
\qquad\qquad\qquad \mathsf{net}[\ (\mu\ z_4)(\ \mathsf{st}(z_2,z_4)\ \mid\ \mathsf{st}(z_4,z_3)\ )\ ](z_2)\ )\ ](y)\end{array}$$

2.3 A Normalised Form for Terms of AGN

The axioms we just presented allow us to standardise the term-like representation of nested graphs, by transforming them into an equivalent normalised form. This form is not unique in general, but the equivalence among terms in this form can be characterised precisely by the existence of a structural bijection among them, as explained below.

Definition 6 (normalised form). *A term $\mathbb{G}$ is in* normalised form *if either it is* $\mathbf{0}$*, or it has the shape*

$$(\nu\,\overline{y})(\mu\,\overline{z})(\ \prod_{i=0}^{n} x_i : s_i \ \mid\ \prod_{j=0}^{m} b_j[\mathbb{G}_j](\overline{y}_j)\ )$$

where $n+m>0$, all nodes in $\overline{y}$ and $\overline{z}$ are pairwise distinct, all terms $\mathbb{G}_j$ for $j \in \underline{m}$ are ν-free and normalised themselves and, letting $X \triangleq \bigcup_{i=1}^{n}\{x_i : s_i\}$, we have $\#X = n$, $fn(\mathbb{G}) \cup |\overline{y}| \cup |\overline{z}| = X$, and $fn(\mathbb{G}_j) = X$ for all $j \in \underline{m}$.

Proposition 1 (normalised form). *For any* **AGN** *term $\mathbb{G}$ it is possible to find a $\equiv_{\mathcal{A}}$-equivalent term $\mathbb{H}$ in normalised form.*

Proof (sketch). Roughly, the normalisation proceeds by first α-renaming all the restricted nodes so to make them pairwise distinct and also distinct from all the free nodes (axiom A7). Then all the restrictions are moved towards the top by applying axioms A4–A6 and A8. Note that while any ν-restriction can reach the top of the term (by A8), μ-restrictions cannot escape from their enclosing edge. Then, axioms A9–A10 are used to "saturate" each subgraph with all nodes available. For this task, we point out that $(\omega\ x : s)\mathbb{G} \equiv_{\mathcal{A}} (\omega\ x : s)(x : s \mid \mathbb{G})$: in fact, this property is trivial if $x : s \in fn(\mathbb{G})$ (by axiom A9), while otherwise it follows from

$$\begin{aligned}
(\omega\ x : s)\mathbb{G} &\equiv_{\mathcal{A}} (\omega\ x : s)(\mathbb{G} \mid \mathbf{0}) && \text{(by axiom A3)}\\
&\equiv_{\mathcal{A}} \mathbb{G} \mid (\omega\ x : s)\mathbf{0} && \text{(by axiom A6)}\\
&\equiv_{\mathcal{A}} \mathbb{G} \mid (\omega\ x : s)x : s && \text{(by axiom A5)}\\
&\equiv_{\mathcal{A}} (\omega\ x : s)(\mathbb{G} \mid x : s) && \text{(by axiom A6)}\\
&\equiv_{\mathcal{A}} (\omega\ x : s)(x : s \mid \mathbb{G}) && \text{(by axiom A1)}
\end{aligned}$$

Finally, we exploit axioms A1–A3 to properly rearrange the order of subgraphs composed in parallel, according to the shape of the normalised form. □

Example 3. The graph in Fig. 1 can be written in normalised form as the **AGN** term $(\nu\ y)(\ x \mid y \mid \mathsf{net}[(\mu\ z_1, z_2, z_3)(\mathbb{D}'_1 \mid \mathbb{G}'_1)](y)\ )$, where

$$\begin{aligned}
&\mathbb{D}'_1 \triangleq x \mid y \mid z_1 \mid z_2 \mid z_3\\
&\mathbb{G}'_1 \triangleq \mathsf{st}[\mathbb{D}'_1](x, y) \mid \mathsf{st}[\mathbb{D}'_1](z_3, z_1) \mid \mathsf{st}[\mathbb{D}'_1](z_1, z_2) \mid \mathsf{st}[\mathbb{D}'_1](z_2, z_3) \mid \mathsf{net}[\mathbb{G}_2](z_2)\\
&\mathbb{G}_2 \triangleq (\mu\ z_4)(\ \mathbb{D}'_2 \mid \mathsf{st}[\mathbb{D}'_2](z_2, z_4) \mid \mathsf{st}[\mathbb{D}'_2](z_4, z_3)\ )\\
&\mathbb{D}'_2 \triangleq x \mid y \mid z_1 \mid z_2 \mid z_3 \mid z_4
\end{aligned}$$

Clearly, the normalised form of a term is not unique, not only because of α-conversion (axiom A7) but also because of the AC axioms for $\mid$ (A1 and A2) and of axiom A4, which allows to switch restrictions of the same type in an arbitrary way; nevertheless, it can be shown that these are the only sources of non-uniqueness. In fact, it is tedious but not difficult to prove that two terms $\mathbb{G}$ and $\mathbb{H}$ in normalised form are equivalent *if and only if* they have the same free nodes, and a suitable partial bijection ϕ can be established between the sets of nodes of their corresponding syntax trees. Quite informally, ϕ must relate nodes with corresponding B-labelled boxes and μ- and ν-restrictions, preserving the nesting w.r.t. B-labelled boxes: essentially it records the permutations that can be applied to μ- and ν-restrictions of $\mathbb{G}$ (using A4) and to B-labelled boxes (using A1 and A2) in order to transform $\mathbb{G}$ into $\mathbb{H}$ (up to α-conversion).

The characterisation of the $\equiv_{\mathcal{A}}$-equivalence by the existence of a partial bijection is exploited in Section 4 when arguing about the completeness of the encoding of nested graphs into term graphs.

3 Term Graphs and GS-Monoidal Theories

This section introduces term graphs over a signature Σ as models of the gs-monoidal theory over Σ, by slightly generalising the main result of [10]: in fact, we shall consider many-sorted signatures instead of standard one-sorted ones.

Term graphs are defined as isomorphism classes of (ranked) directed acyclic graphs. Our main concern here is to stress the underlying algebraic structure, hence the presentation of term graphs slightly departs from the standard definition. With respect to the way term graphs are defined in the seminal paper [2], the main differences consist in the restriction to the acyclic case and the handling of empty nodes. A discussion about the relationship between the categorical and the traditional definition of term graphs can be found in [10].

Definition 7 (signature). *Given a set S of sorts, a* signature *Σ over S is a family $\{\Sigma_{u,s}\}_{u \in S^*, s \in S}$ of sets of operator symbols. For an operator $f \in \Sigma_{u,s}$, we call u its* arity *and s its* coarity*; sometimes we shall denote it as $f \colon u \to s$.*

Definition 8 (labelled graphs). *Let Σ be a signature over a set of sorts S. A* labelled (hyper-)graph *d* (over *Σ) is a tuple $d = \langle N, E, l_N, l_E, src, trg \rangle$, where N is a finite set of* nodes*, E is a finite set of* edges*, $src \colon E \to N^*$, $trg \colon E \to N$ are the* source *and* target connection functions*, and $l_N \colon N \to S$, $l_E \colon E \to \Sigma$ are the* labelling functions*, colouring nodes with sorts and edges with operator symbols. Furthermore, the following conditions must be satisfied*

1. *the connection functions are required to be consistent with the labelling, i.e., for all $e \in E$, $l_E(e) \in \Sigma_{u,s} \Leftrightarrow l_N^*(src(e)) = u \wedge l_N(trg(e)) = s$;*
2. *each node is the target of at most one edge, i.e., for all $e_1, e_2 \in E$, $trg(e_1) = trg(e_2) \Rightarrow e_1 = e_2$.*

A node n is *empty* if there is no edge $e \in E$ such that $n = trg(e)$; we shall denote by $N_\emptyset$ and N_Σ the sets of empty and non-empty nodes, respectively (thus $N = N_\Sigma \uplus N_\emptyset$). A labelled graph d is *discrete* if $E = \emptyset$. A *path* in d is a sequence $\langle n_0, e_0, n_1, \ldots, e_{m-1}, n_m \rangle$, where $m \geq 0$, $n_0, \ldots, n_m \in N$, $e_0, \ldots, e_{m-1} \in E$, and $n_k \in src(e_k)$, $n_{k+1} = trg(e_k)$ for $k \in \{0, \ldots, m-1\}$. The *length* of this path is m, i.e., the number of traversed edges; if $m = 0$, the path is *empty*. A *cycle* is a path like above where $n_0 = n_m$.

Definition 9 (directed acyclic graphs, dags). *A* directed acyclic graph *or* dag *(over Σ) is a labelled graph which does not contain any non-empty cycle.*

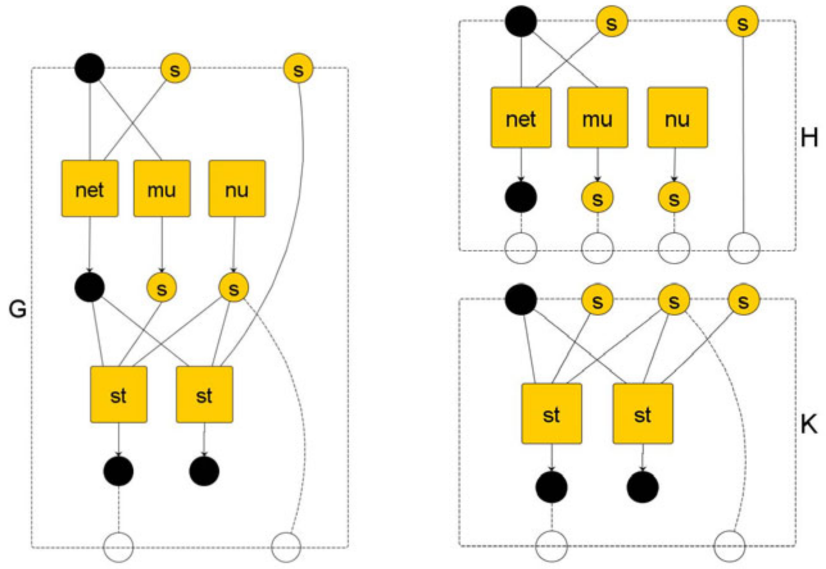


Fig. 5. Some sample term graphs

In the next definition we equip dags with some *attaching points* or *interfaces*, which will be used later to define suitable operations on them. We assume that an arbitrary but fixed signature Σ over a set of sorts S is given.

Definition 10 (ranked dags). *A* (u, w)-ranked dag*, with* $u, w \in S^*$ *(a* dag of rank (u, w)*) is a triple* $g = \langle v, d, r\rangle$*, where* $d = \langle N, E, l_N, l_E, src, trg\rangle$ *is a dag with exactly* $\#u$ *empty nodes,* $v\colon \underline{\#u} \to N_\emptyset$ *is a bijection between* $\underline{\#u}$ *and the empty nodes of* d*, called the* variable mapping*, and satisfying* $l_N(v(j)) = u|_j$ *for all* $j \in \underline{\#u}$*, and* $r\colon \underline{\#w} \to N$ *is a function, called the* root mapping*, satisfying* $l_N(r(i)) = w|_i$ *for all* $i \in \underline{\#w}$*.*

Two (u, w)-ranked dags $g = \langle v, d, r\rangle$ and $g' = \langle v', d', r'\rangle$ are *isomorphic* if there exists a *ranked dag isomorphism* $\phi : g \to g'$ between them, i.e., a pair of bijections $\phi_N\colon N_d \to N_{d'}$ and $\phi_E\colon E_d \to E_{d'}$ preserving connections, labels, roots and variables in the expected way.

Definition 11 (ranked term graphs). *A* (u, w)-ranked term graph *is an isomorphism class* $T = [g]$ *of* (u, w)*-ranked dags. We write* T^u_w *to recall that* T *has rank* (u, w)*.*

Figure 5 illustrates the graphical conventions we use by showing three term graphs over the set of sorts $S = \{\bullet, \mathsf{s}\}$ and the signature $\Sigma = \{\mathsf{nu}\colon \epsilon \to \mathsf{s}, \mathsf{mu}\colon \bullet \to \mathsf{s}, \mathsf{net}\colon \bullet\,\mathsf{s} \to \bullet, \mathsf{st} : \bullet\mathsf{ss} \to \bullet\}$. Nodes are depicted as small circles and edges as rounded boxes, each with the corresponding label inside, but for $\bullet$-labeled nodes which are drawn as black circles. For each edge, the nodes in its source connection are linked with plain lines coming from above and they are ordered from left to right, while a down-going arrow connects an edge to its target node. Clearly, the connection functions are consistent with the labelling; for example each edge labeled by st is linked from above to three nodes labeled $\bullet$, s and s, in this order. By condition 2 of Definition 8 every node has at most one incoming arrow.

The outer dashed rounded boxes conceptually represent the interfaces of the term graphs. The top border, on which all the empty nodes are placed, encodes the variable mapping: an empty node m is in the i-th position (from left to right) if and only if $v(i) = m$. The bottom border depicts instead the root mapping: each "fake" node on it represents an index, and it is connected with a dashed line to its image. Therefore the three term graphs G, H and K have rank $(\bullet\mathsf{ss}, \bullet\mathsf{s})$, $(\bullet\mathsf{ss}, \bullet\mathsf{sss})$ and $(\bullet\mathsf{sss}, \bullet\mathsf{s})$, respectively.

It is fair to notice that these graphical conventions are not standard: in other papers the direction of arrows is reversed, and/or the drawing is flipped vertically. Our choice is consistent with a data-flow interpretation of such graphs, where data flows from top to bottom: every node represents a value that is either produced along the only arrow pointing to it, or that will become available from the environment if the node is empty (and thus it is a variable). Each value "stored" within a node can be used several times along the (possibly dashed) lines that leave downwards. Each edge processes the inputs coming from above and produces one result in its target node. The data-flow orientation is the most appropriate one for presenting the encoding of algebra **AGN** in Section 4, following the intuitive drawing of hierarchical structures.

We introduce now two operations on 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 they correspond in number and sorts. The *union* of term graphs instead is always defined, and it is a kind of disjoint union where roots and variables are suitably concatenated.

Definition 12 (composition and union of ranked term graphs).
Composition. *Let* $T^u_w = [\langle v, d, r\rangle]$ *and* ${T'}^w_z = [\langle v', d', r'\rangle]$ *be ranked term graphs. Their* composition *is the ranked term graph* $S^u_z = T^u_w; {T'}^w_z$ *defined as* $[\langle in_d \circ v, d'', in_{d'} \circ r'\rangle]$*, where* d'' *is obtained from* $d \uplus d'$*, the disjoint union of* d *and* d' *(component-wise on edges and on nodes), modulo the least equivalence relation such that* $r(i) = v'(i)$ *for all* $i \in \underline{\#w}$ *(i.e., by identifying the* i*-th root of* T *with the* i*-th variable of* T'*), and* $in_d, in_{d'}$ *are the inclusions of* d*,* d' *into* d''*.*

Union. *Let* $T^u_w = [\langle v, d, r\rangle]$ *and* ${T'}^x_y = [\langle v', d', r'\rangle]$ *be ranked term graphs. Their* union *or* parallel composition *is the ranked term graph* $S^{ux}_{wy} = T^u_w \otimes {T'}^x_y$ *defined as* $[\langle v'', d \uplus d', r''\rangle]$*, where* $v'' : \underline{\#(ux)} \to N_\emptyset \uplus N'_\emptyset$ *is defined as* $v''(i) = v(i)$ *if* $i \in \underline{\#u}$*, and* $v''(i) = v'(i - \#u)$ *if* $i \in \{\#u + 1, \ldots, \#(ux)\}$*; and* $r'' : \underline{\#(wy)} \to N \uplus N'$ *is defined similarly.*

Figure 5 depicts $G = H; K$, i.e., the term graph $G^{\bullet\mathsf{ss}}_{\bullet\mathsf{s}}$ is the composition of $H^{\bullet\mathsf{ss}}_{\bullet\mathsf{sss}}$ and $K^{\bullet\mathsf{sss}}_{\bullet\mathsf{s}}$. The operations of composition and union on ranked term graphs satisfy various algebraic laws, but we refrain from listing them here because

$$\text{(op)}\ \frac{f \in \Sigma_{u,s}}{f : u \to s} \qquad \text{(id)}\ \frac{u \in S^*}{id_u : u \to u} \qquad \text{(bang)}\ \frac{u \in S^*}{!_u : u \to \epsilon} \qquad \text{(dup)}\ \frac{u \in S^*}{\nabla_u : u \to uu}$$

$$\text{(sym)}\ \frac{u, v \in S^*}{\rho_{u,v} : uv \to vu} \qquad \text{(seq)}\ \frac{t : u \to v \quad t' : v \to w}{t; t' : u \to w} \qquad \text{(par)}\ \frac{t : u \to v \quad t' : u' \to v'}{t \otimes t' : uu' \to vv'}$$

Fig. 6. Inference rules of gs-monoidal theories

they will follow from the result reported in the next section, showing that term graphs form the initial model of gs-monoidal theories (see Theorem 2).

3.1 GS-Monoidal Theories

As anticipated in the Introduction, inspired by the seminal work on flownomial algebras in [12], a sound and complete axiomatisation of ranked term graphs has been proposed in [10]. This result is analogous to the characterisation of (tuples of) terms over a signature Σ as arrows of the Lawvere theory of Σ, considered as the free cartesian category generated by Σ.

However, the categorical framework where such results have been proved is not relevant here, because we are not interested in the details of the proofs, but just in the axiomatisation itself, which allows us to represent every ranked term graph as an expression using suitable operators. The properties of such operators are described by a set of axioms, and the main fact is that equivalence classes of expressions with respect to the axioms are one-to-one with ranked term graphs.

The expressions of interest are generated by the rules depicted in Fig. 6: they are obtained from some basic (families of) terms by closing them with respect to *sequential* (seq) and *parallel* (par) *composition.* By rule (op), the basic terms include one generator for each operator of the signature: these are the elementary bricks of our expressions, and conceptually correspond to the hyper-edges of the term graphs. All other basic terms define the wires that can be used to build our graphs: the *identities* (id), the *dischargers* (bang), the *duplicators* (dup) and the *symmetries* (sym). Every expression $t\colon u \to v$ generated by the inference rules is typed by a *source* and by a *target* sequence of sorts (u and v, respectively), which are relevant only for the sequential composition, which is a partial operation. The next definition presents the axioms imposed on such expressions.

Definition 13 (gs-monoidal theory). *Given a signature Σ over a set of sorts S, the* gs-monoidal theory $\mathbf{GS}(\Sigma)$ *is the category whose objects are the elements of S^* and whose arrows are equivalence classes of* gs-monoidal terms*, i.e., terms generated by the inference rules in Fig. 6 subject to the following conditions*

- *identities and sequential composition satisfy the axioms of categories*
  [identity] $id_u ; t = t = t ; id_v$ *for all* $t : u \to v$*;*
  [associativity] $t_1 ; (t_2 ; t_3) = (t_1 ; t_2) ; t_3$ *whenever any side is defined,*

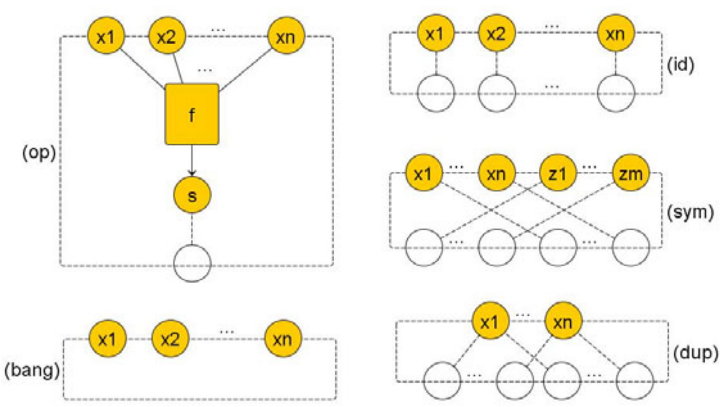


Fig. 7. The term graphs corresponding to the basic arrows of the gs-monoidal theory

– $\otimes$ *is a monoidal functor with unit* id_ϵ*, i.e., it satisfies*
[functoriality] $id_{uv} = id_u \otimes id_v$*, and*
$(t_1 \otimes t_2) ; (t'_1 \otimes t'_2) = (t_1 ; t'_1) \otimes (t_2 ; t'_2)$ *whenever both sides are defined,*
[monoid] $t \otimes id_\epsilon = t = id_\epsilon \otimes t \qquad t_1 \otimes (t_2 \otimes t_3) = (t_1 \otimes t_2) \otimes t_3$
– ρ *is a symmetric monoidal natural transformation, i.e., it satisfies*
[naturality] $(t \otimes t') ; \rho_{v,v'} = \rho_{u,u'} ; (t' \otimes t)$ *for all* $t : u \to v$ *and* $t' : u' \to v'$
[symmetry] $(id_u \otimes \rho_{v,w}) ; (\rho_{u,w} \otimes id_v) = \rho_{u \otimes v,w} \qquad \rho_{u,v} ; \rho_{v,u} = id_{u \otimes v}$
$\rho_{\epsilon,u} = \rho_{u,\epsilon} = id_u$
– ∇ *and* ! *satisfy the following axioms*
[unit] $!_\epsilon = \nabla_\epsilon = id_\epsilon$
[duplication] $\nabla_u ; (id_u \otimes \nabla_u) = \nabla_u ; (\nabla_u \otimes id_u) \qquad \nabla_u ; (id_u \otimes !_u) = id_u$
$\nabla_u ; \rho_{u,u} = \nabla_u$
[monoidality] $\nabla_{uv} ; (id_u \otimes \rho_{v,u} \otimes id_v) = \nabla_u \otimes \nabla_v \qquad !_{uv} = !_u \otimes !_v$

A wiring *is an arrow of* $\mathbf{GS}(\Sigma)$ *which is obtained from the rules of Fig. 6 without using rule (op).*

Notice that the definition of wiring is well-given, because any operator symbol introduced by rule (op) is preserved by all the axioms of the theory.

Given the above definition, the main result of [10] is summarised as follows.

Theorem 2 (axiomatisation of ranked term graphs [10]). *Let* Σ *be a signature over a set of sorts* S *and let* $u, v \in S^*$*. Then there is a bijective correspondence between term graphs over* Σ *of rank* (u, v) *and arrows of the gs-monoidal theory of* Σ*,* $\mathbf{GS}(\Sigma)$*, from* u *to* v*. In particular, wirings from* u *to* v *are in bijective correspondence with discrete term graphs of rank* (u, v)*.*

Just to give a feeling on how the correspondence stated by the theorem works, the term graph denoted by a gs-monoidal term generated by the rules of Fig. 6 can be built by structural induction: Fig. 7 shows the ranked term graphs corresponding to the basic terms introduced by rules (op), (id), (bang), (dup) and (sym), assuming that $u = x_1, x_2, \cdots, x_n$ and $v = z_1, z_2, \cdots, z_m$; instead, rules (seq) and (par) correspond to the operations of composition and of union as introduced in Definition 12. For example, the (equivalence classes of) terms $t_H \triangleq [((\nabla_\bullet \otimes id_s) ; (id_\bullet \otimes \rho_{\bullet,s}) ; (\mathsf{net} \otimes \mathsf{mu} \otimes \mathsf{nu})) \otimes id_s] \colon \bullet\mathsf{ss} \to \bullet\mathsf{sss}$ and $t_K \triangleq [(((\nabla_\bullet \otimes id_s \otimes \nabla_s) ; (id_\bullet \otimes \rho_{\bullet,ss} \otimes \nabla_s)) \otimes id_s) ; (\mathsf{st} \otimes ((id_{\bullet s} \otimes \rho_{s,s}) ; ((\mathsf{st} ; !_\bullet) \otimes id_s)))] \colon \bullet\,\mathsf{sss} \to \bullet\mathsf{s}$ denote, respectively, the term graphs H and K from Fig. 5, while G corresponds to $t_H ; t_K$.

In the following we shall assume that $\otimes$ has precedence over ;, hence in the example above we can write for example $t_H \triangleq [(\nabla_\bullet \otimes id_s ; id_\bullet \otimes \rho_{\bullet,s} ; \mathsf{net} \otimes \mathsf{mu} \otimes \mathsf{nu}) \otimes id_s] \colon \bullet\mathsf{ss} \to \bullet\mathsf{sss}$, omitting several brackets.

An easy corollary of Theorem 2, that we will need later on, states that each wiring of $\mathbf{GS}(\Sigma)$ denotes a suitable sort-preserving function.

Corollary 1. *Let $u, v \in S^*$. Then the wirings of $\mathbf{GS}(\Sigma)$ from u to v are in bijective correspondence with the set of functions $\{k \colon \underline{\#v} \to \underline{\#u} \mid u|_{k(i)} = v|_i$ for all $i \in \underline{\#v}\}$.*

In fact, each wiring from u to v denotes a discrete term graph of rank (u, v), which is uniquely determined by its root function.

4 From AGN Terms to Term Graphs

In this section we define a translation from the terms of the algebra **AGN** introduced in Section 2 to equivalence classes of terms of the gs-monoidal theory of a suitable signature, and therefore, by Theorem 2, to term graphs over that signature. Disregarding for the moment the technical details, the intuition behind the translation is quite simple: the nesting of edges is rendered in a term graph by a tree-like structure made of nodes of a special sort, representing locations; free nodes are mapped to variables; and each restricted node is encoded as the target node of a special constant.

As a first step, we introduce the signature over which the term graphs obtained by the translation of the terms of **AGN** are defined.

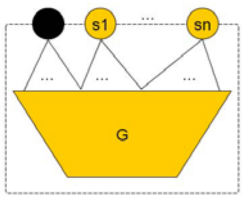
Definition 14 (signature $\Sigma_B^\bullet$). *Given the set of sorts $S^\bullet = S \cup \{\bullet\}$, assuming that $\bullet \notin S$, the signature $\Sigma_B^\bullet$ over $S^\bullet$ is defined as follows*

$$\Sigma_B^\bullet = \{b \colon \bullet, rnk(b) \to \bullet \mid b \in B\} \cup \{\nu_s \colon \epsilon \to s, \mu_s \colon \bullet \to s \mid s \in S\}$$

Intuitively, each box label $b \in B$ corresponds to an operator symbol, having as arity the rank of b preceded by $\bullet$, and $\bullet$ as coarity. The new sort $\bullet$ has a special role in our encoding, because it represents the *locations* in a graph with nesting. An edge labelled with the operator b will be connected (by its first source) to the node representing the location where it lies, and it will “offer” a new location (the target connection), conceptually corresponding to its interior. Furthermore, the signature includes the operator symbols ν_s and μ_s for each sort s: these will be connected through their target connection to the node they restrict; μ_s

additionally has one source of type $\bullet$, which matches with the intuitive definition of "localised restriction".

A term $\mathbb{G}$ of the algebra **AGN** has a *set* of free nodes $fn(\mathbb{G})$ which are used as an interface to the environment, as seen in Section 2. Instead, the "interface" of a term graph is a pair of *lists* of sorts, forming its rank. Each term $\mathbb{G}$ will be translated to a term graph having an empty list of roots, and a linearisation of the free nodes of $\mathbb{G}$ as variables (as exemplified in the figure on the right): therefore the translation is parametric w.r.t. an *assignment*, i.e., a function which assigns a positional index to each free node of the term.



A generic term $\mathbb{G}$ of **AGN** as a term graph (with $fn(\mathbb{G}) = \{x_1 : s_1, \cdots, x_n : s_n\}$)

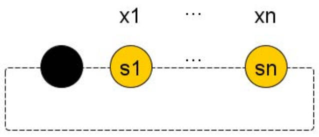
Definition 15 (Assignment). *An* assignment *is a function* $\sigma \in \bigcup_{n\in\mathbb{N}}\{f : \underline{n} \to \mathcal{X}\times S \mid f$ *is injective*$\}$. *An assignment* $\sigma : \underline{n} \to \mathcal{X}\times S$ *for a given* $n \in \mathbb{N}$ *is uniquely determined by a list of nodes without repetitions (because it is injective), namely* $\sigma(1), \sigma(2), \ldots, \sigma(n)$*: we shall often represent it this way and write* $x : s \in \sigma$ *as a shorthand for* $x : s \in img(\sigma)$*, the image of* σ.

In the following, by $\tau(\sigma)$ we denote $\tau(\sigma(1), \sigma(2), \ldots, \sigma(n))$, i.e., the sequence of sorts of the nodes in $img(\sigma)$. Furthermore, for a given list of nodes $\overline{y} \in (\mathcal{X} \times S)^*$ and an assignment σ such that $|\overline{y}| \subseteq img(\sigma)$, we let $k^{\sigma}_{\overline{y}} : \underline{\#\overline{y}} \to \underline{\#\sigma}$ be the function such that $k^{\sigma}_{\overline{y}}(i) = \sigma^{-1}(\overline{y}|_i)$ for all $i \in \underline{\#\overline{y}}$.

Definition 16 (Encoding AGN into gs-monoidal terms). *Let* $\mathbb{G}$ *be a term of* **AGN** *over sorts* S*, box labels* B *and names* $\mathcal{X}$*, and let* $\sigma = x_1 : s_1, \ldots, x_n : s_n$ *be an assignment. We say that* $[\![\mathbb{G}]\!]_\sigma$ *is* well-defined *if* $fn(\mathbb{G}) \subseteq img(\sigma)$*; in this case,* $[\![\mathbb{G}]\!]_\sigma$ *is a term graph of rank* $((\bullet, \tau(\sigma)), \epsilon)$ *over the signature* $\Sigma^\bullet_B$*, defined by structural induction as follows (recall that* $\otimes$ *has precedence over* ;*)*

– $[\![\mathbf{0}]\!]_\sigma = [!_{\bullet,\tau(\sigma)}] : \bullet, \tau(\sigma) \to \epsilon$

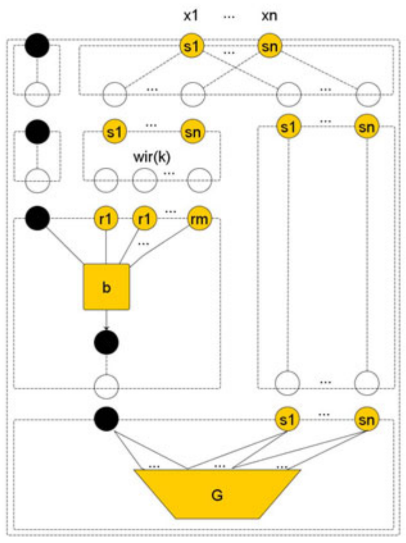
– $[\![x : s]\!]_\sigma = [!_{\bullet,\tau(\sigma)}] : \bullet, \tau(\sigma) \to \epsilon$



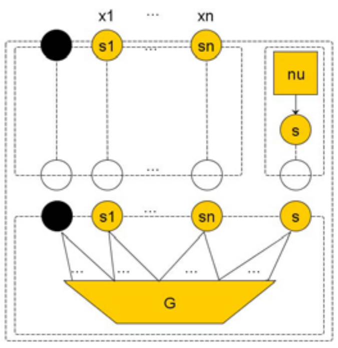
The encodings $[\![0]\!]_\sigma$ *and* $[\![x_i : s_i]\!]_\sigma$*, graphically, assuming* $\sigma = x_1 : s_1, \ldots, x_n : s_n$ *and* $i \in \underline{n}$.

– $[\![b[\mathbb{G}](\overline{y})]\!]_\sigma = [id_\bullet \otimes \nabla_{\tau(\sigma)} ; (id_\bullet \otimes wir(k) ; b) \otimes id_{\tau(\sigma)}] ; [\![\mathbb{G}]\!]_\sigma : \bullet, \tau(\sigma) \to \epsilon$, *where the gs-term* $wir(k) : \tau(\sigma) \to rnk(b)$ *is any representative of the wiring uniquely determined (according to Corollary 1) by the function* $k \triangleq k^{\sigma}_{\overline{y}} : \underline{\#rnk(b)} \to \underline{\#\sigma}$ *defined above (see also Fig. 8)*

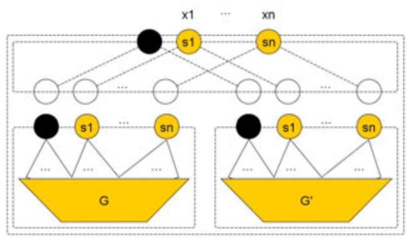
– $[\![\mathbb{G}|\mathbb{G}']\!]_\sigma = [\nabla_{\bullet,\tau(\sigma)}] ; [\![\mathbb{G}]\!]_\sigma \otimes [\![\mathbb{G}']\!]_\sigma : \bullet, \tau(\sigma) \to \epsilon$



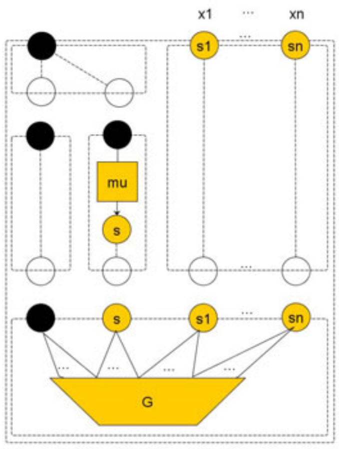
$[\![b[\mathbb{G}](\overline{y})]\!]_\sigma$



$[\![(\nu\ x:s)\mathbb{G}]\!]_\sigma$



$[\![\mathbb{G}|\mathbb{G}']\!]_\sigma$



$[\![(\mu\ x:s)\mathbb{G}]\!]_\sigma$

Fig. 8. The encoding of the terms of algebra **AGN**, graphically

– $[\![(\nu\ x:s)\mathbb{G}]\!]_\sigma = [id_{\bullet,\tau(\sigma)} \otimes \nu_s]\ ;\ [\![\mathbb{G}\{^{y:s}/_{x:s}\}]\!]_{\sigma,y:s} : \bullet, \tau(\sigma) \rightarrow \epsilon$
where $y:s = \mathit{fresh}_{\mathbb{G}}(x:s,\sigma)$

– $[\![(\mu\ x:s)\mathbb{G}]\!]_\sigma = [(\nabla_\bullet\ ;\ id_\bullet \otimes \mu_s) \otimes id_{\tau(\sigma)}]\ ;\ [\![\mathbb{G}\{^{y:s}/_{x:s}\}]\!]_{y:s,\sigma} : \bullet, \tau(\sigma) \rightarrow \epsilon$
where $y:s = \mathit{fresh}_{\mathbb{G}}(x:s,\sigma)$

In the last two rules, $\mathit{fresh}_{\mathbb{G}}(x{:}s,\sigma)$ is a function returning $x{:}s$ itself if $x{:}s \notin \sigma$, and returning a fresh s-sorted node (not appearing neither in σ nor in $\mathbb{G}$) otherwise. Notice that $[\![\mathbf{0}]\!]_\sigma$ and $[\![x:s]\!]_\sigma$ are defined in the same way, but the first is defined for any σ, while the second one is defined only if $x:s \in \sigma$.

Our first main result shows that the encoding is sound w.r.t. the equivalence $\equiv_{\mathcal{A}}$, i.e., that $\equiv_{\mathcal{A}}$-equivalent **AGN** terms are mapped to the same term graph.

Theorem 3 (soundness). *Let $\mathbb{G}$ and $\mathbb{H}$ be two terms of algebra* **AGN** *over sorts S, box labels B and names $\mathcal{X}$ such that $\mathbb{G} \equiv_{\mathcal{A}} \mathbb{H}$. Then, for any assignment σ: 1) $[\![\mathbb{G}]\!]_\sigma$ is well-defined iff $[\![\mathbb{H}]\!]_\sigma$ is such; 2) $[\![\mathbb{G}]\!]_\sigma = [\![\mathbb{H}]\!]_\sigma$ when well-defined.*

Proof (sketch). Item 1) follows by the fact that $\mathbb{G} \equiv_{\mathcal{A}} \mathbb{H}$ implies $fn(\mathbb{G}) = fn(\mathbb{H})$.

To prove item 2), one should show that each axiom A1–A10 from Definition 5 is preserved by the encoding, i.e., that the encoding of the left-hand side of each axiom can be proved equal to the encoding of the right-hand side by exploiting the axioms of gs-monoidal theories (see Definition 13). Note that for axioms A4–A7 one has to consider separately the cases for ν and μ. The detailed proof is omitted for space constraints. □

The second main result of this section states that the encoding is complete w.r.t. the equivalence $\equiv_{\mathcal{A}}$, i.e., that any two **AGN** terms mapped to the same term graph must be $\equiv_{\mathcal{A}}$-equivalent.

Theorem 4 (completeness). *Let $\mathbb{G}$ and $\mathbb{H}$ be two terms of algebra* **AGN** *over node sorts S, box labels B and variables $\mathcal{X}$. If for all assignments σ it holds $[\![\mathbb{G}]\!]_\sigma = [\![\mathbb{H}]\!]_\sigma$, then $\mathbb{G} \equiv_{\mathcal{A}} \mathbb{H}$.*

Proof (sketch). Let us assume, without loss of generality, that terms $\mathbb{G}$ and $\mathbb{H}$ are in normalised form, and that for all assignments σ it holds $[\![\mathbb{G}]\!]_\sigma = [\![\mathbb{H}]\!]_\sigma$. Then they must have the same free nodes, because if $fn(\mathbb{G}) \neq fn(\mathbb{H})$, then it is immediate to find a σ for which only one between $[\![\mathbb{G}]\!]_\sigma$ and $[\![\mathbb{H}]\!]_\sigma$ is defined, contradicting the hypothesis. Next, taken a generic σ such that $fn(\mathbb{G}) \subseteq img(\sigma)$, it can be shown that the rules for the encoding $[\![\_]\!]_\sigma$ induce a suitable partial bijection between the nodes of the syntax tree of $\mathbb{G}$ and the edges of the term graph $[\![\mathbb{G}]\!]_\sigma$ (see also Fig. 8). Since $[\![\mathbb{G}]\!]_\sigma = [\![\mathbb{H}]\!]_\sigma$, these partial bijections can be composed obtaining a partial bijection between the nodes of $\mathbb{G}$ and those of $\mathbb{H}$, which allows us to conclude that they are $\equiv_{\mathcal{A}}$-equivalent, by the considerations at the end of Section 2.3. □

We conclude this section by showing in Fig. 9 the term graph $[\![\mathbb{G}_G]\!]_\sigma$, obtained by applying the encoding of Definition 16 to the NR-graph of Fig. 1 (see Example 2 for the defining expression of $\mathbb{G}_G$), with substitution $\sigma = \{x : s\}$. Because of layout constraints, the term graph is rotated counter-clockwise, exposing the variable mapping on the left border.

5 From Term Graphs to Graphs with Nesting

In this section we prove that there is a one-to-one correspondence between the term graphs obtained by encoding terms of the algebra **AGN** and the NR-graphs introduced in Section 2: this will conclude the proof of Theorem 1.

The first point we address is whether or not the encoding presented in the previous section is surjective, i.e., if any term graph in $\mathbf{GS}(\Sigma_B^\bullet)$ is the image of some term of the algebra **AGN** (for some σ). As our encoding maps to term graphs of rank $(\bullet u, \epsilon)$ only (with $u \in S^*$), the answer is clearly negative in

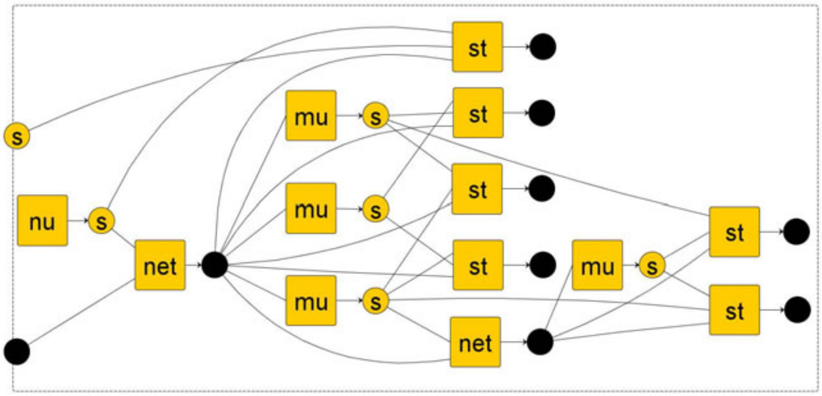
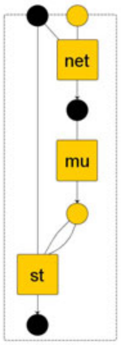


Fig. 9. The term graph $[\![\mathbb{G}_G]\!]_\sigma$ (see Example 2 and Fig. 1)

general. However, even if we restrict to consider term graphs of rank $(\bullet u, \epsilon)$, where $u \in S^*$, the mapping is not surjective. The crucial fact is that the scoping discipline of μ-restriction restricts the visibility of a locally restricted node $x : s$ in such a way that it cannot be used from edges outside the one where $(\mu\ x : s)$ appears, but such a node scoping discipline has no counterpart in term graphs.



Example 4. Let us consider the algebra for our running example of network systems. Then for the term graph $t \triangleq [\nabla_\bullet \otimes id_s \ ; id_\bullet \otimes (\mathsf{net} \ ; \mu_s \ ; \nabla_s) \ ; \mathsf{st} \ ; !_\bullet]$: $\bullet\, \mathsf{s} \to \epsilon$ (see the figure to the right) there is no **AGN** term $\mathbb{G}$ such that $[\![\mathbb{G}]\!]_{x:\mathsf{s}} = t$. In fact, among the natural candidates: the term $\mathsf{net}[(\mu\ y)\mathsf{st}(y, y)](x)$ would be encoded as $[\mathsf{net} \ ; \nabla_\bullet \ ; id_\bullet \otimes (\mu_s \ ; \nabla_s) \ ; \mathsf{st} \ ; !_\bullet]$ with st lying "under" net (and not being a "sibling" of net like in t); the term $\mathsf{net}(x)|(\mu\ y)\mathsf{st}(y, y)$ would be encoded as $[\nabla_\bullet \otimes id_s \ ; (\nabla_\bullet \ ; id_\bullet \otimes (\mu_s \ ; \nabla_s) \ ; \mathsf{st} \ ; !_\bullet) \otimes (\mathsf{net} \ ; !_\bullet)]$ with net and st siblings, but the restriction appearing "outside" net (and not "inside" net, as in t); the term $\mathsf{net}[(\mu\ y)\mathbf{0}](x)|\mathsf{st}(y)$ would have $y : s$ as a free node.

The above counterexample suggests that the algebra **AGN** can serve to characterise exactly those term graphs with well-scoped references to nodes. These are defined as follows.

Definition 17 (well-scoped term graphs). *Let $T = [\langle v, d, r\rangle]$ be a term graph of rank $((\bullet, \tau(\sigma)), \epsilon)$ over the signature $\Sigma_B^\bullet$, with $d = \langle N, E, l_N, l_E, src, trg\rangle$. We say that T is* well-scoped *if for all $e \in E$, for all $n \in src(e)$, if there exists $e' \in E$ such that $n = trg(e')$ and $l_E(e') = \mu$, then $src(e')$ is on a $\bullet$-path (i.e., a path where all nodes are of sort $\bullet$) from $src(e)|_1$ to $v(1)$.*

Informally, this means that in a well-scoped term graph, every edge referring to a locally restricted node n must lie inside the location where n is restricted.

Proposition 2 (AGN terms and well-scoped term graphs). *Given a set of sorts S, a set of ranked labels B and a set of variables $\mathcal{X}$, for each assignment $\sigma = x_1 : s_1, \ldots, x_n : s_n$ there is a one-to-one correspondence between the equivalence classes w.r.t. $\equiv_{\mathcal{A}}$ of* **AGN** *terms with free names in $\{x_1, \ldots, x_n\}$ and well-scoped term graphs of rank $((\bullet, \tau(\sigma)), \epsilon)$ over signature $\Sigma^{\bullet}_B$.*

Proof (sketch). By structural induction it is possible to show that the result of the encoding of Definition 16 is a gs-monoidal term corresponding to a well-scoped term graph; furthermore, by structural induction on the gs-monoidal normal form of well-scoped term graphs, it can be shown that every well-scoped term graph can be obtained as the result of the encoding of a suitable **AGN** term. By Theorem 3 the encoding is consistent with $\equiv_{\mathcal{A}}$-equivalence classes, and by Theorem 4 it is injective on term graphs. $\square$

Well-scoped term graphs can be considered just as an alternative, graphical representation of NR-graphs, where the nesting is represented by a tree of locations, i.e., the $\bullet$-sorted nodes. Formally, this relationship is captured by the next definition and the following result.

Definition 18 (from term graphs to NR-graphs). *Let $T = [\langle v, d, r \rangle]$ be a term graph of rank $((\bullet, \tau(\sigma)), \epsilon)$ over signature $\Sigma^{\bullet}_B$, with $d = \langle N, E, l_N, l_E, src, trg \rangle$.*

For a $\bullet$-sorted node $n \in N$, let $\mathcal{NR}_G(n)$ be the NR-graph defined as $\mathcal{NR}_G(n) = \langle LR, D, l, c, \rho \rangle$, with

- $LR = \{trg(e) : s \mid e \in E \wedge src(e)|_1 = n \wedge l_E(e) = \mu_s\}$
- $D = \{e \in E \mid src(e)|_1 = n \wedge l_E(e) \in B\}$
- $l(e) = l_E(e)$ *for all* $e \in D$
- $c(e) = u$ *when* $src(e) = \bullet u$, *for all* $e \in D$
- $\rho(e) = \mathcal{NR}_G(trg(e))$ *for all* $e \in D$.

Furthermore, let $\mathcal{NR}(T) = \langle FN, GR, \mathcal{NR}_G(v(1)) \rangle$, with

- $FN = \{v(i) : l_N(v(i)) \mid 1 < i \leq \#\tau(\sigma)\}$
- $GR = \{trg(e) : s \mid e \in E \wedge l_E(e) = \nu_s\}$.

Proposition 3 (correctness of the encoding). *In the hypotheses of Definition 18, $\mathcal{NR}(T)$ is a NR-graph if and only if T is well-scoped. Furthermore, the encoding does not depend on the choice of $\langle v, d, r \rangle$ in the equivalence class T (in the sense that the same NR-graph is obtained, up to isomorphism).*

Proof (sketch). By Definition 1 the main fact to prove is that in every NR-graph inside $\mathcal{NR}(T)$ the edges are connected only to available external nodes, i.e., either to global nodes, or to those locally restricted in an enclosing edge. But this is exactly the property ensured by being well-scoped. $\square$

An encoding in the opposite direction, from NR-graphs to well-scoped term graphs, can be defined as well, but it requires more care. In fact, the naive approach of "flattening" the nested structure of the NR-graph by rearranging all its nodes and edges in a single structure might not work, because locally restricted nodes or edges in different sub-graphs could have the same identity. Therefore starting with an NR-graph G, one should first obtain an isomorphic G' with all node and edge identities distinct, and then one can proceed with the flattening. We do not present here the technical details of this construction, but we state its existence, and that it is inverse to $\mathcal{NR}$.

Proposition 4 (from NR-graphs to term graphs). *There exists an encoding $\mathcal{TG}$ such that if σ is an assignment and G is an NR-graph with free nodes in $img(\sigma)$, then $\mathcal{TG}(G, \sigma)$ is a well-scoped term graph of rank $((\bullet, \tau(\sigma)), \epsilon)$. Furthermore, for each well-scoped term graph T of rank $((\bullet, \tau(\sigma)), \epsilon)$, $\mathcal{TG}(\mathcal{NR}(T), \sigma) = T$, and for each NR-graph G with free nodes in $img(\sigma)$, $\mathcal{NR}(\mathcal{TG}(G, \sigma)) \cong G$.*

Note that the equality is strict in $\mathcal{TG}(\mathcal{NR}(T), \sigma) = T$, because the NR-graph $\mathcal{NR}(T)$ obtained from T has all nodes and edges distinct by construction, whereas the equality is only up to isomorphism in $\mathcal{NR}(\mathcal{TG}(G, \sigma)) \cong G$ because $\mathcal{TG}$ may involve the renaming of some nodes and edges.

6 Towards an Enhanced Modelling Framework

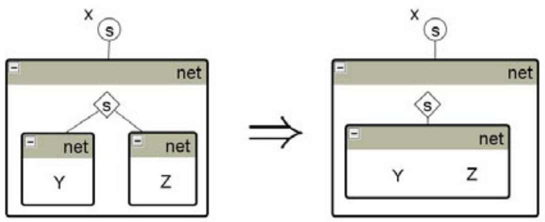
In the previous sections we presented the main technical results, which can be summarised by the following diagram, already presented in the introduction.

$$\begin{tikzcd}
\mathbf{AGN}(S,B)/_{\equiv_{\mathcal{A}}} \arrow[r, Leftrightarrow] \arrow[d, hook, "Sec.\ 4"'] & \text{NR-Graphs over } (S,B) \arrow[d, hook, "Sec.\ 5"] \\
\mathbf{GS}(\Sigma_B^\bullet) \arrow[r, leftrightarrow, "{[10]}"] & \text{Term Graphs over } \Sigma_B^\bullet
\end{tikzcd}$$

Therefore we established a one-to-one correspondence between **AGN** terms up to equivalence, and well-scoped term graphs and NR-graphs up to isomorphism. In this section we first discuss how the relationship with term graphs can be exploited to enrich the visual framework of NR-graphs with a notion of rewriting and with existing analysis techniques. Next we discuss possible generalisations of the proposed framework

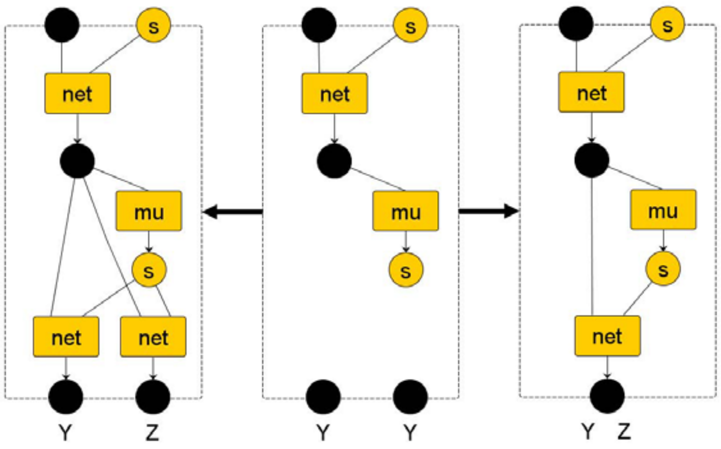
6.1 On Rewriting NR-Graphs

In order to equip our models with behavioural specifications, a natural way is to look for a suitable notion of graph transformation over NR-graphs. For example, one could consider the following transformation rule, intended to model the fact that two local sub-networks can be merged into a single one, which will include the contents of both.



Clearly, one should formalise this notion of rule, as well as its operational meaning, i.e., when it can be applied to a given NR-graph and what the result is. For example, one should clarify the meaning and the role of the variables Y and Z, which are intended to denote the whole content of an edge.

The one-to-one correspondence with term graphs, for which a notion of rewriting is well-understood, can be helpful in this respect. For example, we can translate the left- and the right-hand side of the rule into term graphs, and we can introduce a third term graph in the middle and two morphisms in order to relate the items that have to be preserved, obtaining the *double-pushout* rule



Quite naturally, through this translation the variables correspond to •-labelled nodes, i.e., to locations. This encoding of NR-graph rules into term graph rules can be exploited directly by lifting the definition of term graph rewriting to NR-graphs, or can be used to check the consistency of an original notion of rewriting over NR-graphs. In both cases, it provides a direct link to the rich theory of concurrency and parallelism developed for the algebraic approaches to graph transformation, as well as to the verification techniques developed for them [1]: how far these results can be applied to NR-graphs and their transformations is a subject of future work.

6.2 Edges with Inner Rank: From Term Graphs to GS-Graphs

The distinguishing feature of NR-graphs is the fact that edges are regarded as containers of nested subgraphs. A natural generalisation of this idea would be to equip each edge also with a sorted *inner interface*: while the ordinary "outer interface" is induced by the nodes where the box is attached to, the inner interface would introduce a dual view of local nodes provided by the edge to the nested graph. Such an inner interface could be partly modelled using μ-restriction, but with two main differences: 1) the sorting of the inner interface would be fixed at the signature level, while μ-restricted nodes of any sort can always occur inside a box; 2) the order of nodes provided by the inner interface would be fixed, while μ-restrictions can commute thanks to axiom A4. For example, by equipping each edge with an inner interface having the same rank of the outer interface would provide a straight modelling of modules (each edge) with formal parameters (the nodes provided by the inner interface) and actual parameters (the nodes attached to the outer interface), the correspondence between actual and formal parameters being implicit in the sorting of outer and inner interfaces. Inner interfaces can also be handy for encoding polyadic input prefixes of process calculi, where the input variables are just local place-holders for the values to be received dynamically upon communication.

At the level of **AGN** syntax, this generalisation would correspond to introduce an *inner rank* for each $b \in B$ and to introduce terms like $b[\overline{z}.\mathbb{G}](\overline{y})$, where $\overline{z}$ are the nodes provided by the inner interface of b, whose sorting must match the inner rank of b. At the level of axiomatisation, the nodes provided by the inner interface should be α-convertible ($\overline{z}$ acts as a binder in $b[\overline{z}.\mathbb{G}](\overline{y})$, with scope $\mathbb{G}$) and correspondingly extended versions of axioms A8 and A10 should be given with suitable side-conditions (the notion of free nodes should also be updated).

At the level of the encoding in gs-monoidal theories, this would correspond to move from signatures to *hyper-signatures*, where operators $f \in \Sigma_{u,w}$ with $u, w \in S^*$ are allowed. Interestingly enough, gs-monoidal theories are quite stable and such an extension is seamless, as no additional axiom is required. This is not the case for term graphs, that are tailored to ordinary signatures. This could be annoying, because while we have seen that gs-monoidal theories over ordinary signatures play for term graphs the role played by Lawvere theories for ordinary terms (i.e., they neatly emphasise the essential algebraic structure underlying the set-theoretical presentation of term graphs), the gs-monoidal syntax itself can be hard to follow without the corresponding drawings, even for small terms (see for example Definition 16).

Nevertheless, in the case of hyper-signatures we can resort to consider an alternative model to term graphs, called *gs-graphs* [15], that is defined in terms of concrete (multi-)sets of *assignments*. More precisely, two kinds of assignment are allowed in gs-graphs: a *proper assignment* has the form $x'_1 : s'_1 \dots x'_k : s'_k := f(x_1 : s_1, \dots, x_h : s_h)$ (for $f \in \Sigma_{s_1 \cdots s_h, s'_1 \cdots s'_k}$), while an *auxiliary assignment* has either the form $x' : s := x : s$ (aliasing) or $!(x : s)$ (name disposal). Given a set of assignments A, when a name appears in the left member of an assignment we say that it is *assigned*, when it appears in the right member we say that it is *used* and write $x : s \sqsubset_A x' : s'$ if A contains an assignment where $x : s$ is used and $x' : s'$ is assigned (meaning that, to some extent, $x' : s'$ depends on $x : s$). Like term graphs, also gs-graphs come equipped with top and bottom interfaces: implicitly, the top interface is given by all names that are used but not assigned in A (called *leaves* in [15]), while the bottom interface contains all names $x' : s'$ assigned via an aliasing $x' : s := x : s$ in A (called *roots*). Each interface is ordered according to a fixed total order $\leq$ on sorted names.

A gs-graph A is *valid* if it satisfies all of the following: (1) every name is assigned at most once in A; (2) the transitive closure $\sqsubset_A^+$ of $\sqsubset_A$ is acyclic; (3) every $x' : s$ such that $x' : s := x : s$ belongs to A is a maximal element of $\sqsubset_A^+$;

(4) for each name n not assigned in A (exactly) one disposal $!(n)$ is present in A; (5) for each name n assigned in A no disposal $!(n)$ can be present in A. Then it can be shown that valid gs-graphs on the hyper-signature Σ (taken up to the intuitive notion of isomorphism induced by any injective name substitution that respects the total ordering $\leq$ on the names in the interfaces) are the arrows of the freely generated gs-monoidal category $\mathbf{GS}(\Sigma)$.

In summary, we are confident that the results of the previous sections can be generalised seamlessly by allowing edges with an inner rank in NR-graphs and in the terms of the algebra, and by exploiting gs-graphs rather than term graphs for the encoding.

7 Related Works

As recalled in the Introduction, graphs are widely used in Computer Science for a visual, intuitive representation of systems and models of any kind. Several notions of *hierarchical* graphs have been introduced along the years in various areas, often as a useful structuring mechanism to cope with the modelling of systems of realistic size. One of the earliest proposals are Harel's *higraphs* [18], used first for modelling database structures and next as a basis for statecharts. Several other such models have been proposed since then, for modelling database systems, object-oriented systems and hyper-media applications, among others (see, e.g., the recap in Section 7 of [9]).

In the realm of Graph Transformation Systems, the use of hierarchical graphs dates back to Pratt [21], who used them to represent data structures of programming languages. Several other models have been proposed since them, till the most recent and elaborated ones in [13,9]. The graphs of [13] share with our NR-graphs the fact that subgraphs are encapsulated in (hyper-)edges, but they do not allow arcs to cross edge boundaries. The approach of [9] is instead much more general than ours, because they provide separated representations of a system (given by a "flat" graph) and of its hierarchical structure (an acyclic graph), relating them with a "connection graph". Both these approaches will be sources of inspiration for the definition of graph transformation over our NR-graphs, but none of them provides an algebraic presentation.

More closely related to our proposal, and at the same time direct sources of inspiration for us, are some graph formalisms developed for modelling process calculi. They range from the several approaches based on flat graphs (see, e.g. [16]), with which we share the modelling of name restriction ν, to Milner's *bigraphs* [20]. Basically, a bigraph is given by the superposition of two graphs, representing the *locality* and the *connectivity structure* of a system, respectively, having the nodes in common. In our words, the first specifies the hierarchical structure of the system, while the second the naming topology. So, we do believe that the two approaches have essentially the same expressiveness, even if a precise comparison goes beyond the scope of this paper. It is worth noting, nevertheless, that the two approaches are in a sense dual to each other: bigraphs represent locations of a system as nodes (instead of hyper-edges) and names as hyper-edges (instead of nodes): when designing our modelling framework we preferred to introduce the notion of NR-graph rather than to stick to bigraphs, because NR-graphs allow for a more intuitive representation of systems and have a much simpler definition w.r.t. bigraphs. During the revision of the present paper, we learned that an algebra for bigraphs has been proposed in [17]: we intend to study the precise relationship between this algebra and ours, to understand if the greater complexity of the former is balanced by a greater expressive power. Moreover, the algebra given in [17] is "fine-grained" and closer to the gs-monoidal algebra than to the **AGN** algebra, hence we think the result in this paper is an important step for relating NR-graphs and bigraphs.

Concerning the axiomatisation, several (sound and complete) axiomatisations of various families of graphs exist, and each of them provides a suitable linear syntax for the corresponding graphs. Most of the axiomatisation explicitly address node sharing, possibly following the seminal work on flownomial algebras [12]. It is not possible to mention here all the contributions to this field, but it seems noteworthy that all these structures, including the one discussed in Section 3 and proposed in [10], can be seen as enrichments of symmetric monoidal categories, which thus reasonably provide the basis for the description of distributed environments in terms of wire-and-box diagrams: see the survey [22] and the meta-formalism in [4].

Finally, it is worth stressing that the **AGN** algebra and the corresponding NR-graphs are very close to the algebra introduced in [7], whose semantics is defined set-theoretically over a suitable domain of hierarchical graphs with interfaces. Besides presenting a few technical differences (the hyper-edges of the algebra of [7] also offer an inner interface, like the one discussed in Section 6.2; edges without a nested graph are treated differently; there is only one type of (localised) restriction, and the extrusion of restricted names is handled with optional axioms) a formal encoding of that algebra into term graphs is not available yet, but should not be difficult with the formal background presented here. Instead, that algebra has been used in [5,6,7] to encode several process calculi featuring sophisticated notions of nesting and of restriction (including the π-calculus [19], Sagas [8], and CaSPiS [3], among others).

8 Conclusion and Further Works

In this paper we presented a simple model of hierarchical graphs featuring nesting of subgraphs within hyper-edges and two kinds of restrictions of nodes, which are suited for representing in a direct way a wide class of systems and models. In order to provide a linear, term-like syntax for such graphs, an axiomatisation has been proposed, the main result being that such axiomatisation is sound and complete. The result was proved by encoding such nested structures (both the algebra and the graphs) into a simpler model (term graphs) where the nesting is represented explicitly with a tree of locations, and by exploiting an existing axiomatisation of term graphs as gs-monoidal theories. Finally possible extensions of the presented framework are sketched, including the definition of a notion of rewriting over nested graphs, and the generalisation of the graphical model to allow for edges with inner interfaces.

As topics of future research, besides those just mentioned we intend to clarify the formal relationship between our NR-graphs and Milner's bigraphs [20], and of our algebra with the one recently proposed in [17]. In parallel to this, we intend to test the adequacy of our modelling framework by encoding suitable algebraic formalisms (typically process calculi), which would automatically obtain a graphical representation, as well as visual modelling formalisms, for which we could obtain a handy linear syntax.

References

1. Baldan, P., Corradini, A., König, B.: A framework for the verification of infinite-state graph transformation systems. Information and Computation 206(7), 869–907 (2008)
2. Barendregt, H., van Eekelen, M., Glauert, J., Kennaway, J., Plasmeijer, M., Sleep, M.: Term graph reduction. In: de Bakker, J.W., Nijman, A.J., Treleaven, P.C. (eds.) PARLE 1987. LNCS, vol. 259, pp. 141–158. Springer, Heidelberg (1987)
3. Boreale, M., Bruni, R., Nicola, R.D., Loreti, M.: Sessions and pipelines for structured service programming. In: Barthe, G., de Boer, F.S. (eds.) FMOODS 2008. LNCS, vol. 5051, pp. 19–38. Springer, Heidelberg (2008)
4. Bruni, R., Gadducci, F., Montanari, U.: Normal forms for algebras of connections. Theoretical Computer Science 286, 247–292 (2002)
5. Bruni, R., Corradini, A., Montanari, U.: Modeling a service and session calculus with hierarchical graph transformation (2010) (submitted)
6. Bruni, R., Gadducci, F., Lluch Lafuente, A.: An algebra of hierarchical graphs and its application to structural encoding. Scientific Annals in Computer Science (to appear, 2010)
7. Bruni, R., Gadducci, F., Lluch Lafuente, A.: A graph syntax for processes and services. In: Laneve, C., Su, J. (eds.) Web Services and Formal Methods. LNCS, vol. 6194, pp. 46–60. Springer, Heidelberg (2010)
8. Bruni, R., Melgratti, H.C., Montanari, U.: Theoretical foundations for compensations in flow composition languages. In: Palsberg, J., Abadi, M. (eds.) POPL 2005, pp. 209–220. ACM, New York (2005)
9. Busatto, G., Kreowski, H.J., Kuske, S.: Abstract hierarchical graph transformation. Mathematical Structures in Computer Science 15(4), 773–819 (2005)
10. Corradini, A., Gadducci, F.: An algebraic presentation of term graphs, via gs-monoidal categories. Applied Categorical Structures 7(4), 299–331 (1999)
11. Corradini, A., Montanari, U., Rossi, F.: An abstract machine for concurrent modular systems: CHARM. Theoretical Computer Science 122(1&2), 165–200 (1994)
12. Căzănescu, V.E., Ştefănescu, G.: A general result on abstract flowchart schemes with applications to the study of accessibility, reduction and minimization. Theoretical Computer Science 99(1), 1–63 (1992)
13. Drewes, F., Hoffmann, B., Plump, D.: Hierarchical graph transformation. Journal on Computer and System Sciences 64(2), 249–283 (2002)
14. Ferrari, G.L., Hirsch, D., Lanese, I., Montanari, U., Tuosto, E.: Synchronised hyperedge replacement as a model for service oriented computing. In: de Boer, F.S., Bonsangue, M.M., Graf, S., de Roever, W.-P. (eds.) FMCO 2005. LNCS, vol. 4111, pp. 22–43. Springer, Heidelberg (2006)

15. Ferrari, G.L., Montanari, U.: Tile formats for located and mobile systems. Information and Computation 156(1-2), 173–235 (2000)
16. Gadducci, F.: Graph rewriting for the pi-calculus. Mathematical Structures in Computer Science 17(3), 407–437 (2007)
17. Grohmann, D., Miculan, M.: Graph algebras for bigraphs. In: Ermel, C., de Lara, J., Heckel, R. (eds.) GT-VMT 2010. Electronic Communications of the EASST, vol. 29 (2010)
18. Harel, D.: On visual formalisms. Communication of the ACM 31(5), 514–530 (1988)
19. Milner, R.: Communicating and Mobile Systems. Cambridge University Press, Cambridge (1992)
20. Milner, R.: Pure bigraphs: Structure and dynamics. Information and Computation 204(1), 60–122 (2006)
21. Pratt, T.W.: Definition of programming language semantics using grammars for hierarchical graphs. In: Claus, V., Ehrig, H., Rozenberg, G. (eds.) Graph Grammars 1978. LNCS, vol. 73, pp. 389–400. Springer, Heidelberg (1979)
22. Selinger, P.: A survey of graphical languages for monoidal categories. In: Coecke, B. (ed.) New Structures for Physics. Lecture Notes in Physics. Springer, Heidelberg (to appear, 2010)