Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 253 (2009) 3–15
www.elsevier.com/locate/entcs

A Note on an Old-Fashioned Algebra for (Disconnected) Graphs

Fabio Gadducci[1]

Dipartimento di Informatica, Università di Pisa
Polo Universitario "G. Marconi" di La Spezia

Abstract

Graphs with interfaces are a simple and intuitive tool for allowing a graph G to interact with the environment, by equipping it with two morphisms $J \to G$, $I \to G$. These "handles" were used to define graphical operators, and to provide an inductive presentation of graph rewriting. A main feature of graphs with interfaces is their characterization as terms of a free algebra. So far, this was possible only with discrete interfaces, i.e., containing no edge. This note shows that a similar free construction can be performed also with disconnected interfaces, i.e., containing only nodes connected to at most one edge.

Keywords: Algebraic presentation of graphs, disconnected graphs, DPO approach, parallel derivations.

1 Introduction

Graphs with interfaces equip a (hyper-)graph G with morphisms $j : J \to G$, $i : I \to G$ representing the possible interactions of G with the environment. Indeed, these "handles" were used for defining graphical operators, and exploited for the encoding of process calculi (see e.g. [5] and the references therein) or for providing an inductive presentation of algebraic graph rewriting, notably the DPO approach (see e.g. [1] and the references therein).

Graphs with interfaces have an obvious correspondence with the categorical notion of cospan. As such, they have been often considered as a suitable domain for system specification, e.g. in the work of Walters and others [8]. Most recently, it has been at the basis of the *borrowed context* mechanism [4,11] for equipping graph transformations with a suitable labelled transition system semantics. Also the search for algebraic correspondences between (some sort of) graphs with interfaces and (suitable variants of) monoidal categories –meaning to look for a term-

⋆ This work has been partially supported by the EU within the FP6-IST IP 16004 SEnSOria.

[1] Email: gadducci@di.unipi.it

doi:10.1016/j.entcs.2009.10.014

like presentation of such graphical structures– has been often pursued in categorical terms. This research has quite a long history, at least from the mid-Eighties work of Ştefănescu of others (see the overview offered in [13], and again the references in [1,2]). A recent survey [12] by Selinger offers a concise, yet wide-ranging and uniform illustration of the literature about graphical languages for monoidal categories.

Let us for example take into account the case of possibly cyclic (hyper-)graphs. Quite roughly, adopting a graph transformation jargon, the key reasoning underlying the connection between graphs and categories is plainly told. Consider a signature Σ as a graph (sorts are nodes, operators are edges) and focus on the graphs typed over it, i.e., where nodes (edges) are labelled by sorts (operators, respectively): any such graph with *discrete* interfaces (i.e., such that I, J are just sets of nodes) is uniquely characterized by an arrow of a suitable monoidal category, (almost) freely generated from Σ. See among others [6].

As we wrote, this characterisation allows for providing an inductive presentation of DPO rewriting, and it can be specialised to structures other than graphs (such as e.g. *term graphs*). However, the restriction to discrete interfaces is unfortunate, since it boils down to have rewriting rules with no "read only" component, thus forbidding to properly recast in the algebraic framework the results about parallelism for DPO rewriting.

In this note it is shown how, starting from a type graph U, to define a unary signature Σ_U where the arrows of a suitable *dgs-monoidal* category $\mathbf{DGS}(\Sigma_U)$ correspond to graphs, typed over U, with *disconnected* interfaces, i.e., such that the nodes in I, J are connected to at most one edge. The move from discrete to disconnected interfaces is a pivotal one, since all the concurrency features of the DPO approach are now preserved.

The note is organized as follows. Section 2 recalls the basic notions concerning typed graphs with interfaces; while Section 3 performs the same task concerning dgs-monoidal categories, illustrating their correspondence with graphs with discrete interfaces. Section 4 presents the main remarks about discrete and disconnected graphs; while Section 5 shows some preliminary results on disconnected DPO rewriting. The final Section 6 wraps up this note, offering some further pointers to the literature.

2 Graphs and Graphs with interfaces

This section presents some basic definitions concerning (hyper-)graphs, typed graphs and graphs with interfaces, as well as recalling two operators on graphs with interfaces.

2.1 Typed graphs

We now present the basic notions concerning typed graphs, as discussed e.g. in [3].

Definition 2.1 (graphs) *A graph is a four-tuple $\langle N, E, s, t\rangle$ where N is the set of*

nodes, E is the set of edges and $s, t : E \to N^*$ are the source and target functions.

From now on we denote the components of a graph G by N_G, E_G, s_G and t_G; moreover, for a list S we denote by $|S|$ its underlying set.

Definition 2.2 (two classes of graphs) *A graph G is* discrete *if the set E_G of edges is empty; it is* disconnected *if $s(e)$ and $t(e)$ contain no repeated elements and $|s(e)| \cap |t(e')| = |s(e)| \cap |s(e')| = |t(e)| \cap |t(e')| = \emptyset$ for all edges $e, e' \in E_G$.*

A discrete graph is often going to be presented as just a set.

Definition 2.3 (graph morphisms) *Let G, G' be graphs. A graph morphism $f : G \to G'$ is a pair of functions $\langle f_N, f_E \rangle$ such that $f_N : N_G \to N_{G'}$, $f_E : E_G \to E_{G'}$ and they preserve the source and target functions, i.e., $f_N \circ s_G = s_{G'} \circ f_E$ and $f_N \circ t_G = t_{G'} \circ f_E$.*

The category of graphs is denoted by **Graph**, and similarly for its full subcategories **DGraph** and **IGraph** of disconnected and discrete graphs, respectively.

We now give the definition of typed graph: it can be concisely described as a graph labelled over a structure that is itself a graph.

Definition 2.4 (typed graphs and their morphisms) *Let T be a graph. A* typed graph *G over T is a graph $|G|$ with a graph morphism $\tau_G : |G| \to T$.*

Let G, G' be typed graphs over T. A typed graph morphism *$f : G \to G'$ is a graph morphism $f : |G| \to |G'|$ consistent with the typing, i.e., such that $\tau_G = \tau_{G'} \circ f$.*

The category of graphs typed over T is denoted by T-**Graph**, and similarly for the variants T-**DGraph** and T-**IGraph**. In the following, we assume a fixed type graph T.

2.2 Graphs with interfaces

We are going to need operations to compose graphs. So, we equip typed graphs with suitable "handles" for interacting with an environment, referring to e.g. [1] for an introduction.

Definition 2.5 (graphs with interfaces and their morphisms) *Let J, K be typed graphs. A graph with* input interface *J and* output interface *K is a triple $\mathbb{G} = \langle j, G, k \rangle$, for G a typed graph and $j : J \to G$ and $k : K \to G$ the* input *and* output *typed graph morphisms.*

Let $\mathbb{G}, \mathbb{G}'$ be graphs with the same interface. An interface graph morphism $f : \mathbb{G} \Rightarrow \mathbb{G}'$ is a typed graph morphism $f : G \to G'$ between the underlying typed graphs that preserves the input and output morphisms.

We denote a graph with input interface J and output interface K by $J \xrightarrow{j} G \xleftarrow{k} K$. It is disconnected (discrete) if its interfaces J and K are so. With an abuse of notation, in the following we refer to the items belonging to the image of the input (output) morphism as inputs (outputs, respectively).

In the categorical literature, a graph with interfaces is just what is called a *cospan*. Indeed, the two binary operators on graphs that we introduce below are

motivated by the characterisation of cospans as a suitable monoidal category. We use the graph-inspired terminology since it is more consistent with the development of the following sections.

In order to properly define the operators, though, we need to enforce two constraints. First of all, we assume that a choice for the disjoint union of graphs is fixed (or, equivalently, that the categorical coproduct in **Graph** is chosen). Moreover, we refer implicitly to a graph with interfaces as the representative of its isomorphism class, sometimes denoting the class of isomorphic graphs and its components by the same symbol.

Definition 2.6 (sequential and parallel composition) *Let* $\mathbb{G} = J \xrightarrow{j} G \xleftarrow{k} K$ *and* $\mathbb{G}' = K \xrightarrow{j'} G' \xleftarrow{k'} I$ *be graphs with interfaces. Their sequential composition is the graph with interfaces* $\mathbb{G};\mathbb{G}' = J \xrightarrow{j''} G'' \xleftarrow{k''} I$*, for* G'' *the pushout in* **Graph** *of the arrows* $K \xrightarrow{k} G$ *and* $K \xrightarrow{j'} G'$*, and* j'' *and* k'' *the uniquely induced arrows.*

Let $\mathbb{G} = J \xrightarrow{j} G \xleftarrow{k} K$ *and* $\mathbb{G}' = J' \xrightarrow{j'} G' \xleftarrow{k'} K'$ *be graphs with interfaces. Their parallel composition is the graph with interfaces* $\mathbb{G} \otimes \mathbb{G}' = (J \uplus J') \xrightarrow{j''} G \uplus G' \xleftarrow{k''} (K \uplus K')$*, for* $\uplus$ *is the disjoint union in* **Graph***, and* j'' *and* k'' *the uniquely induced arrows.*

Intuitively, the sequential composition $\mathbb{G};\mathbb{G}'$ is obtained by taking the disjoint union of the graphs underlying $\mathbb{G}$ and $\mathbb{G}'$, and gluing the outputs of $\mathbb{G}$ with the corresponding inputs of $\mathbb{G}'$ (possibly modulo some further item coalescing). The parallel composition $\mathbb{G} \otimes \mathbb{G}'$ is instead obtained by simply taking the disjoint union of the graphs underlying $\mathbb{G}$ and $\mathbb{G}'$. Note that both operations are defined on "concrete" graphs. However, their results do not depend on the choice of the representatives of their isomorphism classes.

Note that the sequential and parallel composition introduced in Definition 2.6 corresponds to the standard composition in the category $Cospan(T\text{-}\mathbf{Graph})$, and to a monoidal operator in that category, respectively. Indeed, exploiting the operators above, the (isomorphic classes of) T-typed graphs with interfaces form a symmetric monoidal category, denoted by $F(T\text{-}\mathbf{Graph})$; and similarly for its full sub-categories $D(T\text{-}\mathbf{Graph})$ and $I(T\text{-}\mathbf{Graph})$ of T-typed graphs with disconnected and discrete interfaces, respectively. For details about the monoidal structure of the category, we refer to e.g. [2, Appendix].

3 Categories with a DGS-Monoidal Structure

This section recalls the definition of dgs-monoidal category, and states the correspondence between the arrows of a dgs-monoidal category and typed graphs with interfaces.

3.1 A class of monoidal categories

Definition 3.1 (DGS-Monoidal categories) *A (strict)* gs-monoidal category $\mathbf{C}$ *is a six-tuple* $\langle \mathbf{C}_0, \otimes, e, \rho, \nabla, ! \rangle$ *such that* $\langle \mathbf{C}_0, \otimes, e, \rho \rangle$ *is a symmetric strict monoidal category and* $!_a : a \to e$, $\nabla_a : a \to a \otimes a$ *are two families of arrows (indexed by objects* $a \in \mathbf{C}_0$*) satisfying* $!_e = \nabla_e = id_e$, *the* coherence *axioms*

$$\nabla_a ; (\nabla_a \otimes id_a) = \nabla_a ; (id_a \otimes \nabla_a) \qquad \nabla_a ; (id_a \otimes !_a) = id_a \qquad \nabla_a ; \rho_{a,a} = \nabla_a$$

and the monoidality *axioms*

$$\nabla_{a \otimes b} ; (id_a \otimes \rho_{b,a} \otimes id_b) = \nabla_a \otimes \nabla_b \qquad !_a \otimes !_b = !_{a \otimes b}$$

A (strict) dgs-monoidal category $\mathbf{C}$ *is an eight-tuple* $\langle \mathbf{C}_0, \otimes, e, \rho, \nabla, !, \Delta, ? \rangle$ *such that both six-tuples* $\langle \mathbf{C}_0, \otimes, e, \rho, \nabla, ! \rangle$ *and* $\langle \mathbf{C}_0^{op}, \otimes, e, \rho, \Delta^{op}, ?^{op} \rangle$ *are gs-monoidal categories*[2] *satisfying the* partition *axioms*

$$\nabla_a ; \Delta_a = id_a \qquad \Delta_a ; \nabla_a = (\nabla_a \otimes id_a) ; (id_a \otimes \Delta_a)$$

A gs-monoidal functor $\langle F, \phi, \phi_e \rangle : \mathbf{C} \to \mathbf{C}'$ *is a symmetric monoidal functor (that is, a functor* F *equipped with two natural isomorphisms* $\phi_e : F(e) \to e'$ *and* $\phi : F(a \otimes b) \to F(a) \otimes' F(b)$*) such that* $F(!_a) ; \phi_e = !'_{F(a)}$ *and* $F(\nabla_a) ; \phi = \nabla'_{F(a)}$*; similarly for* dgs-monoidal functors*, additionally preserving* Δ *and* $?$*. The category of (strict) dgs-monoidal categories and their functors is denoted by* **DGS**.

Mimicking the correspondence between terms and trees, arrows of a free gs-monoidal category correspond to acyclic term graphs (among other venues, see e.g. [1]) and arrows of a free dgs-monoidal category correspond to graphs (see e.g. [6]): in the same way as *terms* over a signature are represented by arrows of its free algebraic theory. As an example, the lack of naturality of morphisms ∇ (i.e., of axioms $s ; \nabla = \nabla ; (s \otimes s)$) allows for the distinction between the sharing of a term and the occurrence of two copies of it.

Now, let us consider again the category of graphs with (discrete/disconnected) interfaces: it is actually dgs-monoidal. Indeed, for each interface X, the morphism ∇_X is represented by the triple $\langle X, X, X \uplus X \rangle$, and the obvious arrows; the morphism $!_X$ is represented by the triple $\langle X, \emptyset, \emptyset \rangle$; and symmetrically for Δ_X and $?_X$. The monoidal structure is not strict, since the parallel composition is e.g. not associative. Now, by abuse of notation, we let $F(T\text{-}\mathbf{Graph})$ ($D(T\text{-}\mathbf{Graph})$ and $I(T\text{-}\mathbf{Graph})$) denote also the monoidal category of graphs with (disconnected and discrete, respectively) interfaces, typed over T, with the additional (non strict) dgs-monoidal structure outlined above.

[2] $\mathbf{C}_0^{op}$ denotes the dual category of $\mathbf{C}_0$, obtained inverting the direction of the arrows, including the families Δ^{op} and $?^{op}$.

3.2 Algebras and categories

We now formally state the correspondence between graphs and dgs-monoidal categories. In different disguises, that result have been often proved, presumably starting from the work by Ştefănescu in the Eighties. We just refer to [1,6] for the precise use of the terminology.

Definition 3.2 ((generalized) signature) *A (generalized) signature* $\Sigma = \langle S, O \rangle$ *is a set* S *of sorts, and a family* $O = \biguplus_{\sigma,\sigma' \in S} O_{\sigma,\sigma'}$ *of operators, for* $\sigma, \sigma' \in S^*$.

In other words, for us a (generalized) signature is nothing more than a (hyper-)graph: sorts are nodes, and operators $o \in O_{\sigma,\sigma'}$ are edges with source and target the tuples $s(o) = \sigma$ and $t(o) = \sigma'$, respectively. We are then going to use the two terms interchangeably.

Consider now the free construction associating to a unary signature Σ (i.e., such that $\sigma, \sigma' \in S$) a dgs-monoidal category $\mathbf{DGS}(\Sigma)$: its objects are tuples of sorts, and its arrows are inductively generated, associating to each operator $o \in \Sigma_{s,s'}$ an arrow $o : s \to s'$, and closing with respect to the axioms and additional arrows required in Definition 3.1.

Proposition 3.3 *Let* Σ *be a unary signature, and let* $\mathbf{DGS}(\Sigma)$ *be the free dgs-monoidal category over* Σ. *Then, there exists a full and faithful dgs-monoidal functor from* $\mathbf{DGS}(\Sigma)$ *to the category* $I(\Sigma\text{-}\mathbf{Graph})$ *of* Σ*-typed graphs with discrete interfaces.*

The result above implies that arrows of $\mathbf{DGS}(\Sigma)$ are in bijective correspondence with Σ-typed graphs with discrete interfaces, as long as a choice of the interfaces is made (meaning that canonical representatives for isomorphic disconnected interfaces are chosen). Most importantly, the proposition states that the isomorphism of these graphs can be recast in equational terms, and thus verified by using the laws holding for dgs-monoidal categories.

3.3 A most basic example

In order to help visualising the correspondence stated in Proposition 3.3 above, consider the graphs depicted in Fig. 1: they are typed over the type graph T_E containing just two edges, labelled g and f, and three different nodes, labelled X, Y, and Z.

Nodes are denoted by their sort. Moreover, nodes in the input (output) interface are denoted by circling them with a dotted (solid, respectively) circle: it might be restrictive (a node might occur twice in an interface), but for this simple example will do. Finally, edges are boxes with an entering tentacle (from the source) and a leaving tentacle (to the target).

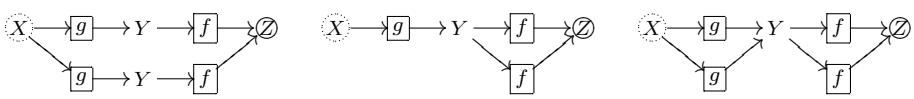

Fig. 1. Graphs with discrete interfaces F_1, F_2, and F_3.

For example, graph F_1 contains four nodes and four edges. The node labelled X is an input node, while Z is an output node. Edges are labelled by boxing their type: the two edges in F_1 labelled f have an outgoing tentacle to the node labelled Z, and an incoming tentacle from two different nodes, both labelled Y.

Considered as a signature, T_E contains three sorts, $\{X, Y, Z\}$; and two operators $g \in O_{X,Y}$ and $f \in O_{Y,Z}$.[3] Now, graphs F_1 and F_2 are the graphical counterparts of the arrows $\nabla_X; (g \otimes g); (f \otimes f); \Delta_Z$ and $g; \nabla_Y; (f \otimes f); \Delta_Z$, respectively, belonging to hom-set $\mathbf{DGS}(\Sigma)[X, Z]$. They are clearly not isomorphic, the obvious difference being the duplication (or lack thereof) of the sub-graph containing the edge g in F_2. This difference is mirrored by the fact that ∇ is not a family of natural transformations, so $g; \nabla_Y$ is not the same as $\nabla_X; (g \otimes g)$. Similarly, graph F_3 is the counterpart of $\nabla_X; (g \otimes g); \Delta_Y; \nabla_Y; (f \otimes f); \Delta_Z$, and it differs from F_1 and F_2 for the amount of sharing of the node Y.

4 An Algebra of Disconnected Graphs

In the previous section we recalled a basic, well-known result concerning graphs with discrete interfaces. We now plan to show how that result can be lifted in order to capture also disconnected interfaces, rounding the section with a simple example.

4.1 From graphs to unary signatures

This section presents the remark motivating the paper, relating disconnected graphs over a type T with free algebras for a derived signature Σ_T.

Definition 4.1 (graph signature) *Let G be a graph. The unary signature Σ_G associated to G has $S = E_G \uplus N_G$ as the set of sorts, and $O = \biguplus_{e \in E_G, n \in N_G} O_{e,n}$ as the family of operators, for $s_i \in O_{e,n}$ ($t_j \in O_{e,n}$) if the i-th element of the list $s(e)$ is n (the j-th element of the list $t(e)$ is n, respectively).*

In other words, the set of sorts is given by the disjoint union of the sets of edges and nodes; while the unary operators mimic the source and target functions of the graph. The solution is indeed old-fashioned: it corresponds to the standard construction of a bipartite graph with unary edges only from an hyper-graph.

Proposition 4.2 (algebras for disconnected graphs) *The category T-**Graph** of T-typed graphs is isomorphic to the category $\mathbf{Alg}_{\Sigma_T}$ of algebras over Σ_T; moreover, its full sub-category T-**DGraph** of T-typed disconnected graphs is isomorphic to the full sub-category of $\mathbf{Alg}_{\Sigma_T}$ of the free algebras over Σ_T.*

The proof is straightforward. Indeed, for the full categories T-**Graph** and $\mathbf{Alg}_{\Sigma_T}$ it is well-known, motivating the interest in unary algebras of e.g. the graph rewriting community. The disconnected case is also easy, since only the sets of edges and nodes are relevant. It has been probably overlooked, since there has been so far little use for such graphs.

[3] Differently from graph rewriting tradition, tentacle direction adopts a "data flow" view instead of a "control flow" one.

The result below is now a consequence of Propositions 3.3 and Proposition 4.2.

Corollary 4.3 *Let* $\mathbf{DGS}(\Sigma_T)$ *be the free dgs-monoidal category over* Σ_T*. Then, there exists a full and faithful dgs-monoidal functor from* $\mathbf{DGS}(\Sigma_T)$ *to the category* $D(T\text{-}\mathbf{Graph})$ *of* T*-typed graphs with disconnected interfaces.*

4.2 Another most basic example

We now address in some detail a very basic example: indeed, the most basic of all of them.

Let us consider the type graph U in Figure 2. It has a unique node, N, and a unique edge, E, and clearly $s(E) = t(E) = N$. As a signature, it just contains a sort N and an operator $E \in O_{N,N}$. The algebras over U are not that interesting. Basically, these are simple graphs (i.e., such that there exists at most one edge between two nodes) additionally verifying that there is exactly one tentacle leaving from each node. Instead, the graphs typed over U are exactly the standard graphs, and the discrete ones are just sets of nodes. As for disconnected graphs, up-to isomorphism they might be simply presented just as a set of edges and a set of *isolated* nodes.

Consider now Σ_U, depicted as a graph on the right of Figure 2: it has two nodes/sorts, namely $\{E, N\}$, and two edges/operators, $s_1, t_1 \in O_{E,N}$; and let G be the U-typed graph on the left of Figure 3 (with respect to Figure 1, for the sake of clarity the items are indexed). The algebra over Σ_U corresponding to G has $X_E = \{E_1, E_2, E_3\}$ and $X_N = \{N_1, N_2, N_3, N_4\}$ as the sets of sorts E and N in the carrier, respectively. Also the operators are intuitively given, with $s_1(E_i) = N_1$ and $t_1(E_i) = N_i$ for $i = 1, 2, 3$.

What is noteworthy is that a disconnected graph is isomorphic to the algebra over Σ_U freely generated from its underlying sets of edges and isolated nodes. Consider e.g. the disconnected graph $d(G)$ on the right of Figure 3: it corresponds to the free algebra over Σ_U with (sorted) sets of variables X_E and X_N. Indeed, it is a sort of disconnected variant of G, as it is made formal in Definition 5.6.

Fig. 2. The type graph U (left) and the associated unary signature Σ_U, as a graph (right).

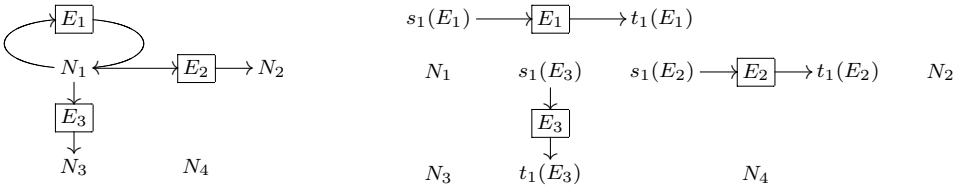

Fig. 3. A U-typed graph G (left) and its disconnected variant $d(G)$ (right).

5 Disconnected Graph Rewriting

It is now the turn to consider graph rewriting, showing how the use of disconnected rules still allows for capturing the concurrency features of the framework.

5.1 Some basics of DPO rewriting

In this section we introduce the basic definitions for the DPO approach to the rewriting of typed (hyper-)graphs [3] and graphs with interfaces.

Definition 5.1 (production) *A T-typed graph production $p : (L \xleftarrow{l} K \xrightarrow{r} R)$ is a production name p and a pair of graph morphisms $l : K \to L$, $r : K \to R$ in* T-**Graph**.

Note that we do not assume that the left-hand side morphism l is injective in T-**Graph**. We say that a production is disconnected (discrete) if the intermediate graph K is so.

Definition 5.2 (derivation) *Let $p : (L \xleftarrow{l} K \xrightarrow{r} R)$ be a T-typed graph production and G a T-typed graph. A match of p in G is a morphism $m_L : L \to G$. A direct derivation from G to H via production p and match m_L is a diagram as depicted in Figure 4, where (1) and (2) are pushouts in* T-**Graph**. *We denote this derivation by $p/m : G \Longrightarrow H$, for $m = \langle m_L, m_K, m_R \rangle$, or simply by $G \Longrightarrow H$.*

$$\begin{array}{cccccc} p: & L & \xleftarrow{l} & K & \xrightarrow{r} & R \\ & {\scriptstyle m_L}\downarrow & (1) & {\scriptstyle m_K}\downarrow & (2) & \downarrow{\scriptstyle m_R} \\ & G & \xleftarrow[l^*]{} & D & \xrightarrow[r^*]{} & H \end{array}$$

Fig. 4. A direct derivation.

Before giving the definition of derivation between graphs with interfaces, we need to introduce the notion of track relation.

Definition 5.3 (track relation) *Let p be a graph production and let $p/m : G \Longrightarrow H$ be a direct derivation, as in Figure 4. Then, the* track relation *$tr(p/m)$ associated with the derivation is the pair of relations (on graphs and nodes) given by $r^* \circ (l^*)^{-1} : G \to H$.*

Let $p/m : G \Longrightarrow H$ be a direct derivation and let F be a sub-graph of G. Then, the track relation $tr(p/m)$ is functional *on F if its restriction $tr(p/m)_{|_F} : F \to H$ to the items of F actually induces a graph morphism.*

Whenever it is functional, the track relation identifies the single items before and after a derivation. It is then used to enforce the preservation of interfaces during a derivation.

Definition 5.4 (derivation between graphs with interfaces) *Let $\mathbb{G} = J \xrightarrow{j} G \xleftarrow{k} K$ and $\mathbb{H} = J \xrightarrow{j'} H \xleftarrow{k'} K$ be graphs with interfaces, and let $p/m : G \Longrightarrow H$ be a direct derivation such that the track relation $tr(p/m)$ is functional on $j(J)$ and*

$k(K)$. We say that $p/m : \mathbb{G} \Longrightarrow \mathbb{H}$ is a direct derivation between graphs with interfaces if $j' = tr(p/m) \circ j$ and $k' = tr(p/m) \circ k$.

Thus, intuitively, a derivation between graphs with interfaces is a direct derivation between the underlying graphs, such that inputs and outputs are preserved.

Finally, we can discuss about the independence of consecutive steps.

Definition 5.5 (sequential independence) *Let $p_1/m_1 : G \Longrightarrow H$ and $p_2/m_2 : H \Longrightarrow I$ be two direct derivations as in Figure 5. Then, these derivations are* sequentially independent *if there exist two graph morphisms $i_1 : R_1 \to D_2$ and $i_2 : L_2 \to D_1$ such that $l_2^* \circ i_1 = m_{R_1}$ and $r_1^* \circ i_2 = m_{L_2}$.*

5.2 On disconnected derivations

It is well-known that, from the point of view of reachability, restricting the attention to discrete productions does not imply losing any generality. This is not the case anymore from the point of view of concurrency, since the items of the intermediate graph represent those items that are preserved (that is, read but not consumed) during a derivation.

Definition 5.6 (disconnecting graphs) *Let G be a T-typed graph. Its disconnected variant $d(G)$ is the disconnected graph associated to the free algebra in Σ_T generated by the sets of edges and nodes of G.*

The definition is well-given, since an algebra in Σ_T uniquely identifies a T-typed graph. So, for a T-typed graph G we denote by $d_G : d(G) \to G$ the obvious graph morphism.

Definition 5.7 (disconnecting interfaces and productions) *Let $\mathbb{G} = J \xrightarrow{j} G \xleftarrow{k} K$ be a graph with interfaces. Then, its disconnected variant is the graph with disconnected interfaces $d(\mathbb{G}) = d(J) \xrightarrow{j \circ d_J} G \xleftarrow{k \circ d_K} d(K)$.*

Let $p : (L \xleftarrow{l} K \xrightarrow{r} R)$ be a T-typed production. Then, its disconnected variant is the disconnected production $d(p) : (L \xleftarrow{l \circ d_K} d(K) \xrightarrow{r \circ d_K} R)$.

So, we can obtain from either a graph or a production a disconnected one, and, most importantly, the same occurs for derivations.

Lemma 5.8 *Let p be a graph production and let $p/m : G \Longrightarrow H$ be a direct derivation, as in Figure 4. Then there is a direct derivation $d(p)/d(m) : G \Longrightarrow H$ with production $d(p)$ and match $d(m) = \langle m_L, m_K \circ d_K, m_r \rangle$.*

The proof of the result above is easily recovered by looking at Figure 6. The derivation $d(p)/d(m)$ is called the disconnected variant of p/m, and denoted as $d(p/m)$.

Corollary 5.9 *Let p be a graph production and let $p/m : \mathbb{G} \Longrightarrow \mathbb{H}$ be a direct derivation between graphs with interfaces. Then there is a direct derivation between graphs with interfaces $d(p/m) : d(\mathbb{G}) \Longrightarrow d(\mathbb{H})$.*

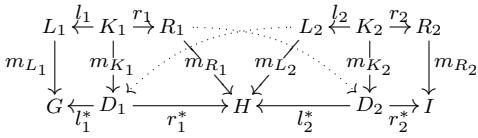

Fig. 5. Sequential independence for $p_1/m_{L_1} : G \Longrightarrow H$ and $p_2/m_{L_2} : H \Longrightarrow I$.

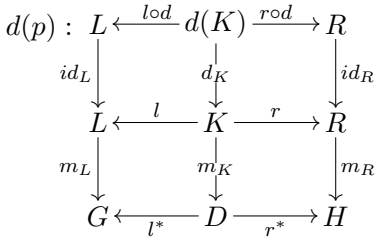

Fig. 6. Disconnected variant of a direct derivation.

The only important remark is to note that $tr(p/m)$ and $tr(d(p/m))$ clearly coincide, and moreover that the image of J (of K) of a graph with interfaces $J \longrightarrow G \longleftarrow K$ characterizes the same sub-graph of G as the image of the interface $d(J)$ (of the interface $d(K)$, respectively) of its disconnected variant, since d_J (d_K, respectively) is surjective.

Please note that more direct derivations can be performed by $d(p)$ than by p. However, the former class could be suitably restrained: chosen a match m_L, a one-to-one correspondence could be obtained, for example by considering only those derivations obtained via the so-called *natural* pushout complement. We refer the reader to [7, Section 3].

From our point of view, it suffices to state the result below.

Proposition 5.10 (disconnection preserves independence) *Let $p_1/m_1 : G \Longrightarrow H$ and $p_2/m_2 : H \Longrightarrow I$ be two direct derivations as in Figure 5. Then, these derivations are sequentially independent if and only if $d(p_1/m_1)$ and $d(p_2/m_2)$ are so.*

6 Conclusions and Further Works

The core of this note is Section 4: it presents the main remarks about the correspondence between discrete and disconnected graphs, showing how to build from a type graph T a category of algebras representing T-typed disconnected graphs. The construction of Σ_T recalls the presentation of hyper-graphs as bipartite graphs. As a paradigmatic example, we focused on graphs typed over a signature U with sort N and unary operator $E \in O_{N,N}$, corresponding to standard graphs: each graph is identified by an algebra over the signature Σ_U, with sorts E, N and operators $s_1, t_1 : E \to N$; and graphs with disconnected interfaces, typed over U, are uniquely characterized by the arrows of $\mathbf{DGS}(\Sigma_U)$.

Related to this, Section 5 discusses preliminary results concerning DPO rewriting with disconnected rules. Disconnected graphs are relevant from the point of

view of graph rewriting, since they allow for an inductive presentation of the direct derivations obtained via the DPO approach, yet preserving all its concurrency features. While the latter issue is discussed with some details in Section 5, we do not tackle the former, and we refer the reader to [6] and especially [7]. Indeed, disconnected graphs were first studied there, and the present paper can be considered as a reformulation and generalization (to hyper graphs) of some of the results occurring there, partly filtered through some remarks in [10].

Indeed, there have been a lot of inspiring works before, even if I sparsely quote other papers here. I apologize in advance, and refer the reader to the references contained in the few items occurring in the bibliography below. Note however that even the same name of dgs-monoidal categories is the variant we currently adopt, and equivalent structures occurred in different disguises in the literature since some time, as e.g. commutative separable algebras [10] or as an instance of (collapsed, with $a^* = a$) compact closed categories [12].

We would like to further investigate the properties of $\mathbf{DGS}(\Sigma_U)$, in the light of the use of adhesive categories as a framework for the generalization of DPO rewriting [9], and possibly for including concurrency in labels distilled using the borrowed context mechanism.

Acknowledgement

I am indebted to Andrea Corradini, Aberto Lluch-Lafuente and Giacoma Valentina Monreale for the interesting discussions and the careful reading of the paper.

References

[1] Corradini, A. and F. Gadducci, *Rewriting on cyclic structures: Equivalence between the operational and the categorical description*, Informatique Théorique et Applications/Theoretical Informatics and Applications **33** (1999), pp. 467–493.

[2] Corradini, A. and F. Gadducci, *A functorial semantics for multi-algebras and partial algebras, with applications to syntax*, Theor. Comp. Sci. **286** (2002), pp. 293–322.

[3] Corradini, A., U. Montanari, F. Rossi, H. Ehrig, R. Heckel and M. Löwe, *Algebraic approaches to graph transformation I: Basic concepts and double pushout approach*, in: G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation, I: Foundations*, World Scientific, 1997 pp. 163–245.

[4] Ehrig, H. and B. König, *Deriving bisimulation congruences in the DPO approach to graph rewriting with borrowed contexts*, Mathematical Structures in Computer Science **16** (2006), pp. 1133–1163.

[5] Gadducci, F., *Graph rewriting for the π-calculus*, Mathematical Structures in Computer Science **17** (2007), pp. 407–437.

[6] Gadducci, F. and R. Heckel, *An inductive view of graph transformation*, in: F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques*, LNCS **1376** (1997), pp. 219–233.

[7] Gadducci, F., R. Heckel and M. Llabrés, *A bi-categorical axiomatisation of concurrent graph rewriting*, in: M. Hofmann, D. Pavlovič and G. Rosolini, editors, *Category Theory and Computer Science*, Electr. Notes in Theor. Comp. Sci. **29** (1999).

[8] Katis, P., N. Sabadini and R. Walters, *SPAN(Graph): A categorical algebra of transition systems*, in: M. Johnson, editor, *Algebraic Methodology and Software Technology*, LNCS **1349** (1997), pp. 307–321.

[9] Lack, S. and P. Sobociński, *Adhesive and quasiadhesive categories*, Informatique Théorique et Applications/Theoretical Informatics and Applications **39** (2005), pp. 511–545.

[10] Rosebrugh, R., N. Sabadini and R. Walters, *Calculating colimits compositionally*, in: P. Degano, R. De Nicola and J. Meseguer, editors, *Concurrency, Graphs and Models*, LNCS **5065** (2009), pp. 581–592.

[11] Sassone, V. and P. Sobociński, *Reactive systems over cospans*, in: *Logic in Computer Science* (2005), pp. 311–320.

[12] Selinger, P., *A survey of graphical languages for monoidal categories*, in: B. Coecke, editor, *New Structures for Physics*, Lecture Notes in Physics (2009), to appear. Available at http://arxiv.org/abs/0908.3347.

[13] Ştefănescu, G., "Network Algebra," Springer, 2000.