

Hierarchical Graph Transformation Revisited

Transformations of Coalgebraic Graphs

Julia Padberg^(✉)

Hamburg University of Applied Sciences, Hamburg, Germany
julia.padberg@haw-hamburg.de

Abstract. Concepts for structuring are fundamental to any modelling technique. Hierarchical graphs allow vertical structuring, where nodes or edges contain other nodes or subgraphs. There have been several suggestions to hierarchical graphs that differ in terms of the underlying graph type, the elements that are structured and the way the structuring is achieved. In this contribution we aim at a more general notion of hierarchical structures for graphs. We investigate several extensions of the powersets that comprise arbitrarily nested subsets, and call them superpower set. This allows the definition of graphs with possibly infinitely nested nodes. Additionally, we allow edges that are incident to edges. Coalgebras and comma categories are used to capture different notions of hierarchies. The main motivation of this paper is the question how to define recursion on a graph's structure so that we still obtain an \mathcal{M} -adhesive transformation system.

Keywords: Graph · Hierarchy · Coalgebra · \mathcal{M} -adhesive transformation system

1 Motivation

Graphs are commonly used to describe complex structures that may become very large. For the sake of scalability many approaches using graphs have one or more additional structuring notions. Hierarchical graphs (and graph transformations) add some hierarchy to the nodes or to the edges. Various approaches to graphs with hierarchy have been proposed, e.g. [5, 7–9, 19, 22, 24, 28, 29]. The resulting techniques were used for modelling hierarchical hypermedia, distributed project management, mobile and ubiquitous systems among others.

In this contribution we investigate the possible variations with respect to their use in graph transformation systems. This allows choosing an adequate hierarchical model and directly obtaining the results from \mathcal{M} -adhesive transformation systems. The concept of graphs is very general and can be specialised to directed and undirected graphs, as well as hypergraphs, typed, labelled or attributed graphs. To cope with that many different approaches \mathcal{M} -adhesive transformation systems offer an abstract definition that requires some categorical constructs and provides a common theory for many different types of graphs and the corresponding transformation systems.

Hierarchies are naturally presented by means of a recursion, since items of one hierarchical level are expanded to several items of the next lower level. In Sect. 2 we represent the recursive structure of the hierarchy as two constructions that allow nested subsets of subsets. This construction – the superpower set – yields the corresponding functors. Subsequently we investigate these functors and show that coalgebraic graphs based on these functors yield \mathcal{M} -adhesive transformation systems. One advantage is that the additional graph structure is not represented by additional arcs but in terms of containment given in a uniform way based on functors. Moreover, we expand the notion of edges so that edges between edges are possible. Although this leads to unconventional concepts such edges have already been used, e.g. in AGG [2, 14] or for graph grouping [19]. We investigate several notions of hierarchies in graphs in more detail in Sect. 5:

Hierarchical Graphs comprise many different approaches to hierarchies in graph, in Sect. 5.1 we discuss the relation to three of them, hierarchical hyperedges as in [9] and packages in [8]. The latter is a more general approach that allows various types of graphs. In [24] hierarchical graphs are given that allow trees as hierarchies for both nodes and edges.

Multi-Hierarchical Graphs as presented in [29] are sketched in Sect. 5.2. Their hierarchy concept is given for nodes and they support several different hierarchies in the same graph that are independent of each other.

Bigraphs [22] are an important theory for modelling ubiquitous computing and combine two graphs to model both locality and connectivity. The first states containment in terms of a forest, the latter the connection via hyperedges. They can be considered to be a special case of hierarchical graphs (see Sect. 5.3).

Graph Grouping as given in [19] allows the analysis of large graphs by grouping nodes and edges into super nodes and super edges, see Sect. 5.4.

The main benefit of this contribution is the systematic representation of various hierarchy concepts for graphs that support graph transformation in terms of \mathcal{M} -adhesive transformation systems, i.e. the algebraic double pushout (DPO) approach, see Sect. 3. Since the hierarchy concepts are the main focus we only have investigated graph types without any additional data as labels or attributes. Section 4 gives an overview over the hierarchy concepts that are obtained as coalgebraic graphs by varying the underlying categorical concepts. It summarizes the concepts at an informal level and facilitate the establishment of transformation systems for non-standard hierarchical graphs. Then we examine in Sect. 5 various different approaches to hierarchies in graphs. This contribution does not aim at a general theory of hierarchical graphs but at the availability of algebraic graph transformations for widely spread graph hierarchy concepts.

Structure of the paper. In the next section (Sect. 2) we investigate the different constructions for an iterated power set construction. Section 3 details the use of coalgebras for the construction of \mathcal{M} -adhesive categories. The emerging hierarchical concepts are sketched in Sect. 4. We then discuss various approaches to hierarchical graphs (see Sect. 5) and exemplarily compare them to their formulation in terms of \mathcal{M} -adhesive categories in Sect. 4. Subsequently in Sect. 6, we discuss related work and concluding remarks in Sect. 7 close this contribution.

2 Extension of the Powerset

One of the main technical results of this contribution is given in this section. We introduce superpower sets¹ that allow arbitrary nesting of subsets and are the basic construction for the hierarchy concepts.

The superpower set is achieved by recursively inserting subsets of the superpower set into itself. We here present two possibilities \mathbb{P} and \mathcal{P}^ω . Both are functors that preserve pullbacks of injective morphisms and hence can be deployed in the formulation of \mathcal{M} -adhesive transformation systems. These can be constructed from existing ones by various categorical constructions, namely product, coslice, functor and comma categories (see Theorem 4.15 (construction of (weak) adhesive HLR categories) in [10]) and from F -coalgebras based on suitable functors F (see Theorem 1). In Sect. 4 we summarize the potential constructions arising from these results.

The superpower set construction can be defined in various ways, here we present two of them (see the supplemental report [23] for additional ones). Both allow nesting of subsets of arbitrary depth.

Definition 1 (Superpower set \mathbb{P}). *Given a set M and $\mathcal{P}(M)$ the power set of M then we define the superpower set $\mathbb{P}(M)$:*

1. $M \subset \mathbb{P}(M)$ and $\mathcal{P}(M) \subset \mathbb{P}(M)$.
2. If $M' \subset \mathbb{P}(M)$ then $M' \in \mathbb{P}(M)$.

$\mathbb{P}(M)$ is the smallest set satisfying 1. and 2.

Condition 1 ensures that sets may contain nested subset of different depth. \mathcal{P}^ω differs from \mathbb{P} as in each subset there are only subsets that have the same depth in terms of nesting. So, for some non-empty set M with $m \in M$ we have $\{m, M\} \notin \mathcal{P}^\omega(M)$ but $\{m, M\} \in \mathbb{P}(M)$.

Definition 2 Superpower set \mathcal{P}^ω . *Given a set M we define $\mathcal{P}^0(M) = M$ and $\mathcal{P}^1(M) = \mathcal{P}(M)$ the power set of M . Then $\mathcal{P}^{i+1}(M) = \mathcal{P}(\mathcal{P}^i(M))$ and $\mathcal{P}^\omega(M) = \bigcup_{i \in \mathbb{N}_0} \mathcal{P}^i(M)$.*

The use of strict subsets ensures in both definitions that Russell's antinomy cannot occur. Both superpower set constructions yield well-founded sets with an order based on the depth of the nested parentheses and hence allow induction.

Subsequently, we only investigate the properties of \mathbb{P} . The results hold for \mathcal{P}^ω and other possibilities of superpower sets as well, see [23].

Lemma 1 (\mathbb{P} is a functor). $\mathbb{P} : \mathbf{Sets} \rightarrow \mathbf{Sets}$ is defined for sets as in Definition 1 and for functions $f : M \rightarrow N$ by $\mathbb{P}(f) : \mathbb{P}(M) \rightarrow \mathbb{P}(N)$ with

$$\mathbb{P}(f)(x) = \begin{cases} f(x) & ; x \in M \\ \{\mathbb{P}(f)(x') \mid x' \in x\} & ; \text{else} \end{cases}$$

¹ Although the nesting of sets or nodes (see Sect. 6) are well-known, to the author's knowledge neither the construction nor the corresponding functors have been considered before.

Example 1 (Functor \mathbb{P}). Given sets $N_1 = \{u, v, w\}$, $N_2 = \{n_1, n_2, n_3\}$ and $f : N_1 \rightarrow N_2$ with $f : u \mapsto n_1, v \mapsto n_1, w \mapsto n_3$ then we have $\mathbb{P}(f) : \mathbb{P}(N_1) \rightarrow \mathbb{P}(N_2)$ with for example $\mathbb{P}(f)(\{u, v, w, \{u, v\}, \{v, \{w, \emptyset\}\}\}) = \{n_1, n_3, \{n_1\}, \{n_1, \{n_3, \emptyset\}\}\}$.

Lemma 2 (\mathbb{P} preserves injections). *Given injective function $f : M \rightarrow N$ then $\mathbb{P}(f) : \mathbb{P}(M) \rightarrow \mathbb{P}(N)$ is injective.*

For the proof see Lemma 2 in [23]. In this paper the proofs are only given if they are relevant for understanding the concepts.

Lemma 3 (\mathbb{P} preserves pullbacks along injective morphisms). *Given a pullback diagram (PB) in **Sets** with injective $g_1 : C \hookrightarrow D$, then $\mathbb{P}(A)$ in the diagram (1) is pullback in **Sets** as well.*

Proof. Pullbacks and the superpower set functor (see

Lemma 2) preserve injections, so $\pi_B : A \hookrightarrow B$, $\mathbb{P}(\pi_B) : \mathbb{P}(A) \hookrightarrow \mathbb{P}(B)$ and $\pi_{\mathbb{P}(B)} : P \hookrightarrow \mathbb{P}(B)$ are injective. Since (PB) is a pullback diagram we have $A = \{(b, c) \mid f_1(b) = g_1(c)\}$. Since (1) commutes, since \mathbb{P} is a functor. Let P be the pullback of $(\mathbb{P}(D), \mathbb{P}(f_1), \mathbb{P}(g_1))$, so $\mathbb{P}(f_1) \circ \pi_{\mathbb{P}(B)} = \pi_{\mathbb{P}(C)} \circ \mathbb{P}(g_1)$. Hence, $P = \{(B', C') \mid \mathbb{P}(f_1)(B') = \mathbb{P}(g_1)(C')\} \subseteq \mathbb{P}(B) \times \mathbb{P}(C)$.

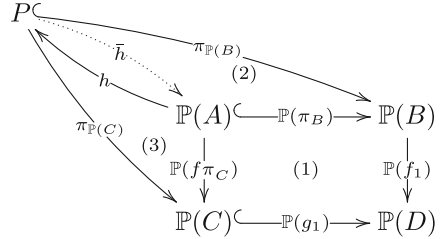
Moreover, there is the unique $h : \mathbb{P}(A) \rightarrow P$ s.t. $h(A') = (\mathbb{P}(\pi_B)(A'), \mathbb{P}(f\pi_C)(A'))$ for all $A' \subseteq A$ so that the diagrams (2) and (3) commute along h : $\pi_{\mathbb{P}(B)} \circ h = \mathbb{P}(\pi_B)$ and $\pi_{\mathbb{P}(C)} \circ h = \mathbb{P}(f\pi_C)$.

We define $\bar{h} : P \rightarrow \mathbb{P}(A)$ with

$$\bar{h}((X, Y)) = \begin{cases} (b, c) ; & \text{if } X = b \in B, Y = c \in C \\ \{(x, y) \mid x \in X \cap B, y \in Y \cap C, f_1(x) = g_1(y)\} \cup \\ & \bigcup_{(X', Y') \in (X-B) \times (Y-C)} \bar{h}(X', Y') ; & \text{else} \end{cases}$$

and have:

1. \bar{h} is well-defined since $\bar{h}(X, Y) \in \mathbb{P}(A)$.
2. (2) commutes along \bar{h} , i.e. $\mathbb{P}(\pi_B) \circ \bar{h} = \pi_{\mathbb{P}(B)}$ is shown by induction over the depth of the nested parentheses n . For $n = 0$, i.e. the atomic nodes, let $(b, c) \in P$ with $b \in B$ and $c \in C$ be given. $\mathbb{P}(\pi_B) \circ \bar{h}(b, c) = \mathbb{P}(\pi_B)(b, c) = b = \pi_{\mathbb{P}(B)}(b, c)$. Let be $\mathbb{P}(\pi_B) \circ \bar{h}(X, Y) = \pi_{\mathbb{P}(B)}(X, Y)$ for sets with at most n nested parentheses. Given $(\hat{X}, \hat{Y}) \in P$ with $n + 1$ nested parentheses and let $\hat{X} = \hat{B} \cup X$ with $\hat{B} \subseteq B$ and $X \cap B = \emptyset$. Let $\hat{Y} = \hat{C} \cup Y$ with $\hat{C} \subseteq C$ and $Y \cap C = \emptyset$. X and Y have at most n nested parentheses. Then



$$\begin{aligned}
& \mathbb{P}(\pi_B) \circ \bar{h}(\hat{X}, \hat{Y}) \\
&= \{x \mid x \in \hat{B} \wedge \exists y \in \hat{C} : f_1(x) = g_1(y)\} \cup \bigcup_{(X', Y') \in (X \times Y)} \mathbb{P}(\pi_B) \circ \bar{h}(X', Y') \\
&\stackrel{IB}{=} \hat{B} \cup \bigcup_{(X', Y') \in (X \times Y)} \pi_{\mathbb{P}(B)}(X', Y') \\
&= \hat{B} \cup X \\
&= \pi_{\mathbb{P}(B)}(\hat{X}, \hat{Y})
\end{aligned}$$

$\mathbb{P}(A)$ is isomorphic to the pullback P , since

- $h \circ \bar{h} = id_P$, since $\pi_{\mathbb{P}(B)}$ is injective and $\pi_{\mathbb{P}(B)} \circ h \circ \bar{h} = \mathbb{P}(\pi_B) \circ \bar{h} = \pi_{\mathbb{P}(B)} \circ id_P$.
- $\bar{h} \circ h = id_{\mathbb{P}(A)}$, since $\mathbb{P}(\pi_B)$ is injective and $\mathbb{P}(\pi_B) \circ \bar{h} \circ h = \pi_{\mathbb{P}(B)} \circ h = \mathbb{P}(\pi_B) = \mathbb{P}(\pi_B) \circ id_{\mathbb{P}(A)}$.

For a first impression graphs with arbitrarily nested nodes are defined. Note, only the nodes are nested, but nodes containing others do not have a name. In Fig. 1 the node $\{m_1, m_2, m_3\}$ contains the nodes m_1, m_2 and m_3 but it does not have a name. Edges are hyperedges given as a subset of the superpower set, so the first level of the nesting of subsets defines the edges, so the edge a_4 connects the nodes $\{n_1, n_2\}$ and $\{n_2, n_4\}$. The edges have neighbours that are atomic nodes or nodes containing nodes and are given by the neighbour function $ngb : E \rightarrow \mathbb{P}(N)$.

The category of \mathbb{P} -graphs is given by a comma category $\langle Id_{\mathbf{Sets}} \downarrow \mathbb{P} \rangle$. The morphisms are given by mappings of the nodes and arcs $f = (f_N, f_E) : G_1 \rightarrow G_2$ with $f_N : N_1 \rightarrow N_2$ and $f_E : E_1 \rightarrow E_2$ so that (1) commutes, i.e. $\mathbb{P}(f_N) \circ ngb_1 = ngb_2 \circ f_E$.

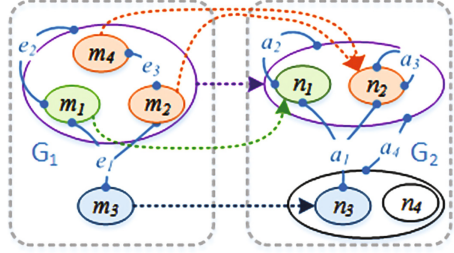


Fig. 1. Morphism in $\langle Id_{\mathbf{Sets}} \downarrow \mathbb{P} \rangle$

$$\begin{array}{ccc}
E_1 & \xrightarrow{ngb_1} & \mathbb{P}(N_1) \\
f_E \downarrow & (1) & \downarrow \mathbb{P}(f_N) \\
E_2 & \xrightarrow{ngb_2} & \mathbb{P}(N_2)
\end{array}$$

Example 2 (\mathbb{P} -Graph). Figure 1 illustrates two \mathbb{P} -graphs and the morphism based on the mappings $f_N : N_1 \rightarrow N_2$ and $f_E : E_1 \rightarrow E_2$ with $f_N(m_4) = n_2$, $f_N(m_i) = n_i$ and $f_E(e_i) = a_i$ for $i = 1, 2, 3$. So, $\mathbb{P}(f_N)(\{m_1, m_2, m_4\}) = \{n_1, n_2\}$.

$$\begin{array}{ll}
G_1 = (N_1, E_1, ngb_1 : E_1 \rightarrow \mathbb{P}(N_1)) \text{ with } & (N_2, E_2, ngb_2 : E_2 \rightarrow \mathbb{P}(N_2)) \\
N_1 = \{m_1, \dots, m_4\} & \text{and } G_2 = \text{ with } N_2 = \{n_1, \dots, n_4\} \\
E_1 = \{e_1, e_2, e_3\} & E_2 = \{a_1, \dots, a_4\} \\
ngb_1 : e_1 \mapsto \{m_1, m_2, m_3\} & ngb_2 : a_1 \mapsto \{n_1, n_2, n_3\} \\
e_2 \mapsto \{m_1, \{m_1, m_2, m_4\}\} & a_2 \mapsto \{n_1, \{n_1, n_2\}\} \\
e_3 \mapsto \{m_2, m_4\} & a_3 \mapsto \{n_2\} \\
& a_4 \mapsto \{\{n_1, n_2\}, \{n_2, n_4\}\}
\end{array}$$

3 \mathcal{M} -Adhesive Categories Using Coalgebras

To express that nodes contain nodes again we need a mapping of nodes to superpower set of nodes $\text{cnt} : N \rightarrow \mathbb{P}(N)$. This is essentially a coalgebra. Coalgebras are often used for specifying the behaviour of systems and data structures that are potentially infinite, for example classes in object-oriented programming, streams and transition systems.

The second main result shows that coalgebras of functors preserving pullbacks along injective morphisms form an \mathcal{M} -adhesive category.

An endofunctor $F : \mathbf{Sets} \rightarrow \mathbf{Sets}$ gives rise the category of coalgebras \mathbf{Sets}_F with $M \xrightarrow{\alpha_M} F(M)$ – also denoted by (M, α_M) – being the objects and morphisms $f : (M, \alpha_M) \rightarrow (N, \alpha_N)$ – called F -homomorphism – so that (1) commutes in \mathbf{Sets} (see [26]).

$$\begin{array}{ccc} M & \xrightarrow{\alpha_M} & F(M) \\ f \downarrow & (1) & \downarrow F(f) \\ N & \xrightarrow{\alpha_N} & F(N) \end{array}$$

Lemma 4 (Pullbacks along injections in \mathbf{Sets}_F). *Given a functor $F : \mathbf{Sets} \rightarrow \mathbf{Sets}$ that preserves pullbacks along an injective morphism, then \mathbf{Sets}_F has pullbacks along an injective F -homomorphism.*

For the proof see Lemma 10 in [23].

\mathcal{M} -adhesive transformation systems (e.g. [12, 25]) are an abstract framework for graph transformations and allow a uniform description of the different notions and results based on a class \mathcal{M} of specific monomorphisms. These well-known results comprise concepts for transformation, local confluence and parallelism, application conditions, amalgamation and so on.

Definition 3 (Class of monomorphisms \mathcal{M}). *Let \mathcal{M} be a class of monomorphisms in \mathbf{Sets} that is PO-PB-compatible, that is:*

1. *Pushouts along \mathcal{M} -morphisms exist and \mathcal{M} is stable under pushouts.*
2. *Pullbacks along \mathcal{M} -morphisms exist and \mathcal{M} is stable under pullbacks.*
3. *\mathcal{M} contains all identities and is closed under composition.*

According to Property 4.7 in [26] if $f : M \rightarrow N$ is injective in \mathbf{Sets} then f is an F -monomorphism in \mathbf{Sets}_F . Obviously the class of all injective functions $\mathcal{M}_F = \{(A, \alpha_A) \xrightarrow{f} (B, \alpha_B) \mid f \text{ is injective in } \mathbf{Sets}\}$ is PO-PB-compatible.

Theorem 1 (\mathcal{M} -Adhesive Category). *$(\mathbf{Sets}_F, \mathcal{M}_F)$ is an \mathcal{M} -adhesive category if F preserves pullbacks along injective morphisms.*

For the proof see proof of Theorem 1 in [23].

This allows \mathcal{M} -adhesive transformation systems for various dynamic systems based on coalgebras of functors the preserve pullbacks along injective morphisms. For a first discussion see [23].

Example 3 (Nested nodes). Nested nodes can be constructed using the coalgebra $\mathbf{Coalg}_{\mathbb{P}}$ based on the superpower set functor \mathbb{P} . Given a set N the function $\text{cnt} : N \rightarrow \mathbb{P}(N)$ gives the nodes contained in a given node. This function yields an \mathcal{M} -adhesive category; the category of coalgebras $\mathbf{Coalg}_{\mathbb{P}}$ over $\mathbb{P} : \mathbf{Sets} \rightarrow \mathbf{Sets}$

with the class \mathcal{M} of injective morphisms.

The nesting of nodes can also be defined allowing the different kinds of nesting using some functor $F : \mathbf{Sets} \rightarrow \mathbf{Sets}$, so we have the contains function $cnt : N \rightarrow F(N)$. This yields an \mathcal{M} -adhesive category where G may be one of the (super-)power functors, e.g. \mathcal{P} , $\mathcal{P}^{(1,2)}$, \mathbb{P} or \mathcal{P}^ω or any other functor preserving pullbacks of injections.

We now investigate the nesting of edges, that is neighbours of edges can again be edges. Edges between edges have been already mentioned for the AGG approach [14] and are used as super edges in graph grouping [19]. To define nested edges we need to extend the neighbour function ngb to edges and nodes. So, we have coalgebraic graphs with directed edges that can be considered as a many sorted coalgebra using the functor $F : \mathbf{Sets} \times \mathbf{Sets} \rightarrow \mathbf{Sets} \times \mathbf{Sets}$ with $F(N, E) = (N, E) \xrightarrow{(!, ngb)} (1, N \times N)$ where 1 is the final object and $!$ the corresponding final morphism, see e.g. [26]. Corollary 2 in [23] extends the result of Theorem 1 to many sorted coalgebras.

Corollary 1. *Let $\mathbb{F} : \mathbf{Sets} \times \mathbf{Sets} \rightarrow \mathbf{Sets} \times \mathbf{Sets}$ be given where \mathbb{F} preserves pullbacks along injections and let \mathcal{M} be the class of pairs of injective morphisms $\langle f_N, f_E \rangle$. Then the category of coalgebraic graphs $(\mathbf{Coalg}_{\mathbb{F}}, \mathcal{M})$ is an \mathcal{M} -adhesive category.*

To give an example we define nested hyperedges.

Example 4 (Nested hyperedges). Given a set of nodes N and a set of edge names E and a function yielding the neighbours $ngb : E \rightarrow \mathcal{P}(V \uplus E)$.

Then the category of coalgebras \mathbf{Coalg}_{F_1} over $F_1 : \mathbf{Sets} \times \mathbf{Sets} \rightarrow \mathbf{Sets} \times \mathbf{Sets}$ with $F_1(N, E) = (1, \mathcal{P}(N \uplus E))$ yields the category of graphs with nested hyperedges. The category of coalgebraic graphs with nested hyperedges \mathbf{Coalg}_{F_1} is an \mathcal{M} -adhesive category. Analogously, graphs with nested, undirected edges can be defined by the functor $\mathcal{P}^{(1,2)}$ that yields only subsets containing one or two elements. And graphs with nested, directed edges can be defined by a functor yielding $(N \uplus E) \times (N \uplus E)$, for both see [23].

Subsequently some properties are sketched that might be worthwhile when investigating coalgebraic graphs more deeply. These properties can be defined independently of the underlying constructs and functors. Their definition depends on the purpose of the approach and we merely hint at a possible formulation as this contribution does not aim at a general theory of hierarchical graph transformation.

Properties of nested nodes can be defined for example as follows:

- Nodes names are unique if cnt is injective.
- Nodes referring to themselves are the atomic nodes, so we define the set $aN = \{n \mid cnt(n) = n\}$.
- Nodes are containers if they are not atomic.
- The set of nodes is well-founded if and only if
 - $X \in N \wedge Y \in cnt(X)$ implies, that $Y \in cnt(N)$

- $X \in \text{cnt}(N) \wedge Y \in (X - N)$ implies, that $Y \in \text{cnt}(N)$
- This ensures that the contains function cnt yields a directed acyclic graph.
- The set of nodes is hierarchical if and only if $\text{cnt}(n) \cap \text{cnt}(n') \neq \emptyset$ implies $n = n'$. This ensures that the neighbour function cnt yields a forest.

Properties of nested edges can be defined for example as follows:

- Common edges are those without nested edges. Hence, the set of common edges is given by those edges e , where $\text{ngb}(e)$ does not contain any edge.
- The function $\text{ngb}^* : E \rightarrow \mathcal{P}(N)$ yields the set of all incident nodes of arbitrarily deep nesting and is defined by $\text{ngb}^*(e) = \{n \in N \mid n \in \text{ngb}(e)\} \cup \bigcup_{x \in \text{ngb}(e)} \text{ngb}^*(x)$. Edges are node-based if ngb^* is well-defined. They are not node-based if they are incident to some container node which is not well-founded.
- Edges are atomic if they are node-based and if the function $\text{ngb}^*(E) \subseteq aN$ only yields atomic nodes.

The notion of subgraphs given by a set of nodes can be transferred easily to the above concepts using the recursive extension $\text{cnt}^*(n) = \{n \in N \mid n \in \text{cnt}(n)\} \cup \bigcup_{x \in \text{cnt}(n)} \text{cnt}^*(x)$. A subgraph $G[M] \subseteq G = (N, E, \text{cnt}, \text{ngb})$ induced by a subset of nodes $M \subseteq N$ can be defined by $G[M] = (M, \{e \in E \mid \text{ngb}(e) \subseteq \text{cnt}^*(M)\}, \text{cnt}, \text{ngb})$ where cnt, ngb are the corresponding restrictions. This yields a notion of subgraph that comprises edges between inducing nodes. Note that in [24] a different approach is chosen (see Sect. 5.1) where edges between nodes that have the same parent need not have this parent.

4 \mathcal{M} -Adhesive Categories of Hierarchical Graphs

The results of the both previous sections yield different \mathcal{M} -adhesive categories depending on the choice of the categorical construction and the underlying functors. We obtain different types of hierarchical graphs and corresponding \mathcal{M} -adhesive transformation systems. Hence (DPO) transformations for each of these types of hierarchical graphs are provided.

The neighbours of edges are given by the neighbour function ngb and the nodes contained by node are given by the contains function cnt . The nodes may be containers or atomic nodes, containers may have a name or not, depending on the construction. The edges can be the well-known ones, (un-)directed, hyperedges with or without an order as well as ones having edges between edges. Next, we state the categorical constructions, involved functors and relate the resulting categories to the examples in this paper. For details and proofs see [23].

Example 5 (Types of hierarchical graphs and corresponding \mathcal{M} -adhesive transformation systems).

1. The comma category $\langle Id_{\mathbf{Sets}} \downarrow \mathbb{P} \rangle$ as used in Example 2 is an \mathcal{M} -adhesive category because of the comma-category construction (see Theorem 4.15 in [10]) and \mathbb{P} preserving pullbacks of injections. It yields hierarchical graphs with hyperedges between nodes and containers of nodes, but containers do not have an explicit name.

2. Combining the nested nodes based on the superpower set functor \mathbb{P} as in Example 3 with usual edges concepts leads to various types of coalgebraic graphs and is closely related to hierarchical graphs in the sense of [8]. In this case hierarchical graphs are given by $G = (N, E, cnt : N \rightarrow \mathbb{P}(N), ngb : E \rightarrow \mathbb{H}(N))$. \mathbb{H} determines edge type. Typical choices for \mathbb{H} are \mathcal{P} or $(_)^*$ for hyperedges, $\mathcal{P}^{(1,2)}$ for undirected edges or for directed edges the copying functor $X^2 : \mathbf{Sets} \rightarrow \mathbf{Sets} \times \mathbf{Sets}$ with $X^2(N) = N \times N$. For an example see Sect. 5.1. Item 1.

We use a coalgebra over $\mathbb{F}_1 : \mathbf{Sets} \times \mathbf{Sets} \rightarrow \mathbf{Sets} \times \mathbf{Sets}$ with $\mathbb{F}(N, E) = \mathbb{P}(N) \times \mathbb{H}(N)$. Then the category of coalgebraic graphs $(\mathbf{Coalg}_{\mathbb{F}_1}, \mathcal{M})$ is an \mathcal{M} -adhesive category.

3. A hierarchy where the edges are refined by subnets (see [9]) is obtained by the neighbouring function $ngb : E \rightarrow (N)^* \times \mathcal{P}^\omega(N)$ that maps edges to a pair where the first component defines the incident nodes and the second component defines the nodes contained by the edges. This nesting is layered as it is defined by the functor \mathcal{P}^ω , see Definition 2. The resulting graphs are given by $G = (N, E, ngb : E \rightarrow N^* \times \mathcal{P}^\omega(N))$. The category of such graphs is given by the comma category $\langle Id_{\mathbf{Sets}} \downarrow \mathbb{G} \rangle$ with the functor $\mathbb{G} = ((_)^* \times \mathcal{P}^\omega) \circ X^2$.

Note $\mathbb{G}(N) = ((_)^* \times \mathcal{P}^\omega) \circ X^2(N) = ((_)^* \times \mathcal{P}^\omega)(N, N) = N^* \times \mathcal{P}^\omega(N)$. For an example see Sect. 5.1. Item 2.

4. For hierarchies where the edges between nodes may have other parents than the nodes and where the edges may contain subgraphs (as in [24]) the coalgebraic graphs are given by the functions $cnt : N \rightarrow \mathbb{P}(N \uplus E)$ and $ngb : E \rightarrow \mathcal{P}(N) \times \mathbb{P}(N \uplus E)$. We use the coalgebra with $\mathbb{F}_2(N, E) = (\mathbb{P}(N \uplus E), \mathcal{P}(N) \times \mathbb{P}(N \uplus E))$. $\mathbb{P}(N \uplus E)$ yields nested sets of nodes and edges and $\mathcal{P}(N)$ yields the incident nodes of an hyperedge. To obtain an \mathcal{M} -adhesive category $\mathbf{Coalg}_{\mathbb{F}_2}$ we construct \mathbb{F}_2 from other functors that yield \mathcal{M} -adhesive categories. An example is in Sect. 5.1. Item 3
5. Multiple hierarchies can be constructed as coalgebraic graphs using a copying functor $X^i : \mathbf{Sets} \rightarrow \mathbf{Sets} \times \mathbf{Sets} \times \dots \times \mathbf{Sets}$. Then the contains function $cnt : N \rightarrow \prod \circ X^i \circ \mathbb{P}(N)$ yields for each node i different nestings. For edges we may use hyperedges $ngb : E \rightarrow \mathcal{P}(N)$. The corresponding \mathcal{M} -adhesive category $\mathbf{Coalg}_{\mathbb{F}_3}$ of coalgebraic graphs is given by $\mathbb{F}_3(N, E) = (\prod \circ X^i \circ \mathbb{P}(N), \mathcal{P}(N))$ and corresponds to the multi-hierarchical graphs in Sect. 5.2.
6. For bigraphs, see Sect. 5.3, we use the functions $cnt : N \rightarrow \mathbb{P}(N)$ and $ngb : E \rightarrow \mathcal{P}(N \uplus E) \times \mathcal{P}(N \uplus E)$. Again we obtain an \mathcal{M} -adhesive category $\mathbf{Coalg}_{\mathbb{F}_4}$ of coalgebraic graphs with $\mathbb{F}_4(N, E) = (\mathbb{P}(N), \mathcal{P}(N \uplus E) \times \mathcal{P}(N \uplus E))$ constructed from other functors.
7. The functions $cnt : N \rightarrow \mathbb{P}(N)$ and $ngb : E \rightarrow N \times N \times \mathbb{P}(E)$ allow the description of graph grouping and give rise to the category of coalgebraic graphs $\mathbf{Coalg}_{\mathbb{F}_5}$ with $\mathbb{F}_5(N, E) = (\mathbb{P}(N), N \times N \times \mathbb{P}(E))$ that corresponds roughly to the the graph grouping in Sect. 5.4.

Varying the constructions, mainly comma categories, product categories and coalgebras and varying the involved functors yield a huge amount of different hierarchy concepts that all lead to well-defined transformation systems. The above examples have been selected to show the width of this approach and to relate it to existing notions of hierarchical graphs. It may as well be used to define new appropriate hierarchy concepts. So, for a specific application the employed hierarchy concept can be chosen out of many different ones.

5 Transformations of Hierarchical Graphs

Here we argue to what extent known concepts can be considered as \mathcal{M} -adhesive categories of hierarchical graphs. The detailed, mathematical investigation of each of these examples is beyond the scope of this paper.

Labels and attributes are not considered in this paper, but labelled or attributed graphs yield \mathcal{M} -adhesive categories (see [10,12]) and at least labels can be introduced into coalgebraic constructions (see [1,26]).

5.1 Hierarchical Graphs

Many possibilities to define hierarchical graphs have already been investigated, e.g. [4,6–9,24]. In [13] the possibility of infinitely recursive hierarchies has already been introduced as an infinite number of type layers. Here we sketch how three of them, namely [8,9,24], can be considered in this framework.

1. Hierarchical Graphs as in [8]

In this approach graphs are grouped into packages via a coupling graph. A hierarchical graph is a system $H = (G, D, B)$, where G is a graph some graph type, P is a rooted directed acyclic graph, and B is a bipartite coupling graph whose partition contains the nodes of N_G and of N_P . All edges are oriented from the first N_G to the second set of nodes N_P and every node in N_G is connected to at least one node in N_P . For this approach we can consider coalgebraic graphs in the coalgebra category $\mathbf{Coalg}_{\mathbb{F}_1}$ (see Example 5.2) with $\text{cnt} : N \rightarrow \mathbb{P}(N)$ being well-founded. Additionally a completeness condition, stating that each atomic node is within some package, has to hold:

$$\forall n \in N : \text{cnt}(n) = n \Rightarrow \exists p \in N : n \in \text{cnt}(p)$$

The packages are the nodes that are not atomic. The edge function is given by $\text{ngb} : E \rightarrow \mathbf{H}(N)$ where $\mathbf{H}(N)$ determines the type of the underlying graphs. In Fig. 2 we have an example with two packages, that uses directed edges. So based on $\mathbf{H} = \mathbf{X}^2$ we can give this example as a coalgebraic graph.

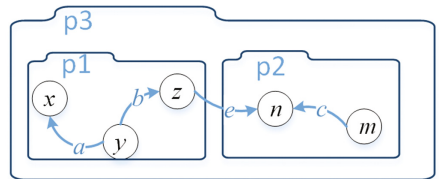


Fig. 2. Hierarchical graph as in [8]

We have $N = \{n, m, x, y, z, p1, p2, p3\}$

$$\text{with } cnt(v) = \begin{cases} v & ; \text{ if } v \in \{n, m, x, y, z\} \\ \{x, y, z\} & ; \text{ if } v = p1 \\ \{n, m\} & ; \text{ if } v = p2 \\ \{p1, p2\} & ; \text{ if } v = p3 \end{cases} \quad \text{and } ngb : \begin{cases} a \mapsto (y, x) \\ b \mapsto (y, z) \\ c \mapsto (m, n) \\ e \mapsto (z, n) \end{cases}$$

2. *Hierarchical Hypergraphs as in [9]* Hypergraphs $H = (V, E, att, lab)$ in [9] consist of two finite sets V and E of vertices and hyperedges. These are equipped with an order, so the attachment function is defined by $att : E \rightarrow V^*$. The hierarchy is given in layers, in the sense that subsets in the same layer have the same nesting depth. So, edges are within one layer. Hierarchical graphs $\langle G, F, cts : F \rightarrow \mathcal{H} \rangle \in \mathcal{H}$ are given with special edges F that contain potentially hierarchical subgraphs. Figure 3a depicts a hierarchical graph that can be considered to be a graph in the comma category $\langle Id_{\mathbf{Sets}} \downarrow \mathbf{G} \rangle$ (see Example 5.3). The graph $G = (N, E, ngb)$ with $ngb : E \rightarrow N^* \times \mathcal{P}^\omega(N)$ is defined so that edges are node-based.
3. *Hierarchical Graphs as in [24]* are obtained from hypergraphs by adding a parent assigning function to them. Nodes and edges can be assigned as a child of any other node or edge. These correspond to coalgebraic graphs in the category $\mathbf{Coalg}_{\mathbb{F}_2}$ (see Example 5.4). The parent function coincides with $cnt : N \rightarrow \mathbb{P}(N \uplus E)$ being well-founded and hierarchical and $ngb : E \rightarrow \mathcal{P}(N) \times \mathbb{P}(N \uplus E)$ since edges can have children as well.

In Fig. 4 the nodes $N = \{1, 2, 3, 6, 8, 9, 11\}$ and the contains function $cnt : N \rightarrow \mathbb{P}(N \uplus E)$, yield the nodes and their children. The hyperedges $E = \{4, 5, 7, 10\}$ with $ngb : E \rightarrow \mathcal{P}(N \times \mathbb{P}(N \uplus E))$ yield the edges. Note in this example the edges are not nested.

Contains and neighbour function are given by

$$\begin{array}{ll} cnt : 1 \mapsto 1 & 8 \mapsto 8 \\ 2 \mapsto 2 & 9 \mapsto 9 \\ 3 \mapsto \{1, 2, 4\} & 11 \mapsto \{8, 9\} \\ 6 \mapsto 6 & \end{array} \quad \text{and } ngb : \begin{array}{l} 4 \mapsto (\{1, 2\}, \emptyset) \\ 5 \mapsto (\{2, 6\}, \emptyset) \\ 7 \mapsto (\{3, 6, 11\}, \emptyset) \\ 10 \mapsto (\{8, 9\}, \emptyset) \end{array}$$

5.2 Multi-Hierarchical Graphs

In [29] multiple hierarchies have been suggested, first ideas can be found in [24]. A finite set of child nesting functions is specified that relate nodes to set of nodes and edges. This corresponds to a finite family $(cnt_i : N \rightarrow \mathbb{P}(N \uplus E))_{i < n}$ that are well-founded and hierarchical. For transformations of multi-hierarchical graphs there is the \mathcal{M} -adhesive category $\mathbf{Coalg}_{\mathbb{F}_3}$ of coalgebraic graphs (see Example 5.5).

5.3 Bigraphs as an Hierarchy

Bigraphs [22] originate in process calculi for concurrent systems and provide a graphical model of computation. A bigraph is composed of two graphs: a place

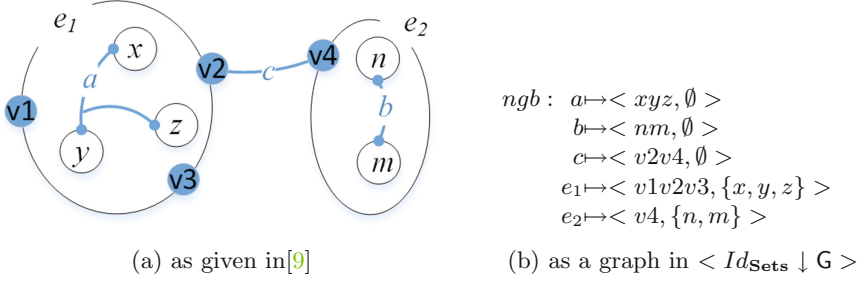


Fig. 3. Example of hierarchical hypergraphs

graph and a link graph. They emphasize interplay between physical locality and virtual connectivity. Reaction rules allow the reconfiguration of bigraphs. A bigraphical reactive system consists of a set of bigraphs and a set of reaction rules, which can be used to reconfigure the set of bigraphs. Bigraphs may be composed and have a bisimulation that is a congruence wrt. composition. Categorically, bigraphs are given as morphisms in a symmetric partial monoidal category where the objects are interfaces. This construction corresponds to ranked graphs as given in [15] where morphisms are given by an isomorphism class of concrete directed graphs with interfaces. [11] discusses extensively the relation of bigraphs to graph transformations. In [16] a functor that flattens bigraphs into ranked graphs is provided that encodes the topological structure of the place graph into the node names. In [5] bigraphs are shown to be essentially the same as gs-graphs that present the place and the link graph within one graph. We also represent bigraphs within one graph, where the hierarchical structure is given by a superpower set of nodes and the link structure is given by nested hyperedges. Here we abstract from the categorical foundations and give bigraphs as a special cases of hierarchical graphs. Hence, we ignore their categorical structure,

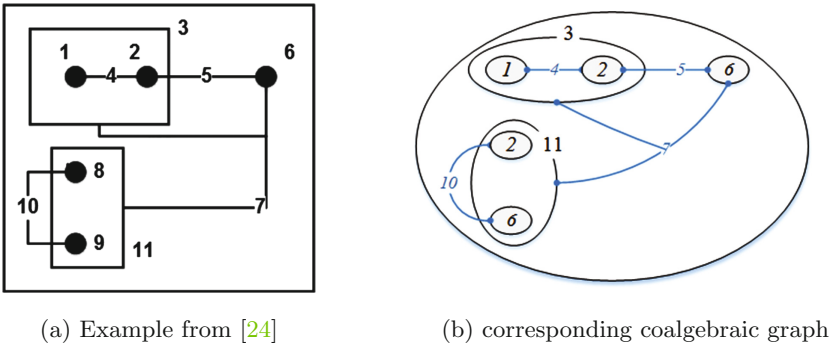


Fig. 4. Hierarchical graph in [24]

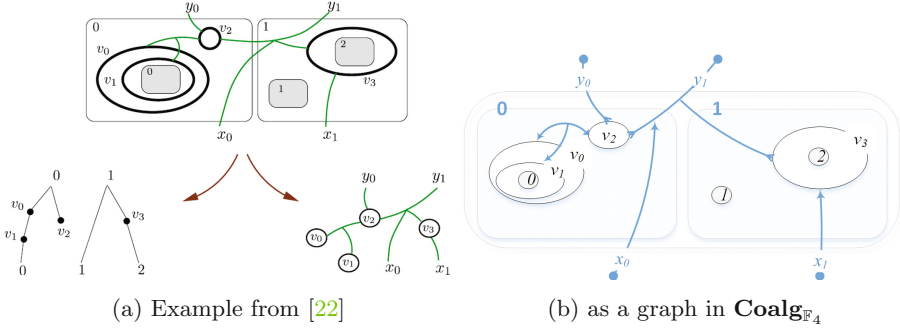


Fig. 5. Bigraph

but we obtain a transformation system. Nevertheless, often only the graphical representation of bigraphs is used [3, 30, 31].

A bigraph is a 5-tuple: $(V, E, ctrl, prnt, link) : \langle k, X \rangle \rightarrow \langle m, Y \rangle$, where V is a set of nodes, E is a set of edges, $ctrl$ is the control map that assigns controls to nodes, $prnt$ is the parent map that defines the nesting of nodes, and $link$ is the link map that defines the link structure. The notation $\langle k, X \rangle \rightarrow \langle m, Y \rangle$ indicates that the bigraph has k holes (sites) and a set of inner names X and m regions, with a set of outer names Y . These are respectively known as the inner and outer interfaces of the bigraph.

Below we illustrate the relation of bigraphs to coalgebraic graphs in $\mathbf{Coalg}_{\mathbb{F}_4}$ (see Example 5.6) in an example. In Fig. 5a we have an introductory example from [22] that we represent as a coalgebraic graph. The coalgebraic graphs in the \mathcal{M} -adhesive category $\mathbf{Coalg}_{\mathbb{F}_4}$ need to have well-founded and hierarchical nodes, where the contains function represents the parent function, so $cnt = prnt$. The function $ctrl$ yields basically the in- and out-degree of each node. The $link$ function yields hyperedges, which we represent as directed hyperedges. Hyperedges connecting outer names are represented as directed hyperedges with the arc itself as the target, those connecting inner names as directed hyperedges with the arc itself as the source. The regions correspond to the roots $0, 1$ of the forests given by cnt and the site are the distinguished atomic nodes $0, 1, 2$. The nodes $N = \{0, 1, v_0, v_1, v_2, v_3, 0, 1, 2\}$ and the contains function $cnt : N \rightarrow \mathbb{P}(N)$, yield the place graph. The directed nested hyperedges $E = \{e_1, e_2, e_3, e_4, e_5\}$ with $ngb : E \rightarrow \mathcal{P}(N \uplus E) \times \mathcal{P}(N \uplus E)$ yield the link graph. We have:

$$\begin{aligned}
 cnt : \quad & 0 \mapsto \{v_0, v_2\} & \text{and } ngb : \quad & e_1 \mapsto (\{v_1, v_2, v_3\}, \{v_1, v_2, v_3\}) \\
 & 1 \mapsto \{v_3, 1\} & & y_0 \mapsto (\{v_2\}, \{v_2\}) \\
 & v_0 \mapsto \{v_1\} & & y_1 \mapsto (\{v_2, v_3\}, \{v_2, v_3\}) \\
 & v_1 \mapsto \{0\} & & x_0 \mapsto (\{x_0\}, \{y_1\}) \\
 & v_2 \mapsto v_2 & & x_1 \mapsto (\{x_1\}, \{v_3\}) \\
 & v_3 \mapsto \{2\} \\
 & i \mapsto i; \text{for } 0 \leq i \leq 2
 \end{aligned}$$

Assuming cnt to be just well-founded we obtain bigraphs with sharing as in [28].

5.4 Graph Grouping

[19] aims at a fundamentally different application area, namely graph grouping to support data analysts making decisions based on very large graphs. Here, a graph hierarchy is established to cope with large amounts of data and to aggregate them. Graph grouping operators produce a so-called summary graph containing super vertices and super edges. A super vertex stores the properties representing the group of nodes, and a super edge stores the properties representing the group of edges. Basically this leads to a contains function $cnt : N \rightarrow \mathbb{P}(N)$ that are well-founded but not necessarily hierarchical and a neighbour function $ngb : E \rightarrow N \times N \times \mathbb{P}(E)$. These can be given as coalgebraic graphs in the category of coalgebras $\mathbf{Coalg}_{\mathbb{F}_5}$ (see Example 5.7) that is \mathcal{M} -adhesive.

But clearly this graph grouping is only sensible for attributed graphs since these used to abstract the data.

6 Related Work

Abstraction in graph transformations is employed for different purposes, e.g. for model checking, for a common theory for different types of graphs, for transferring concepts and results. Abstract approaches to graph transformations [10, 12] of different types of graphs comprise mainly \mathcal{M} -adhesive transformation systems. Other approaches to abstract graphs can be found in that uses the presentation of graphs as a comma-category [17, 27] or as a coalgebra [18]. F -graphs are a family of graph categories induced by a comma category construction using a functor F (see [27]). In [17] the notion of F -graphs based on a construction that is a comma-category and has been encoded as a coalgebraic construction in [18] using the basic idea from [26].

In [20, 21] coalgebraic signatures are used to define various graph types and yields first steps towards a new paradigm for graph transformation systems. Moreover [20] is concerned with attributed graph transformations, since the coalgebraic definition allows a uniform treatment of term algebras over arbitrary signatures and unstructured label sets. in contrast this contribution is concerned with the inner structure of hierarchical graphs and establishes coalgebras as another possible construction for \mathcal{M} -adhesive transformation systems.

7 Concluding Remarks

We have presented a novel approach to hierarchies in graphs and graph transformations. This approach supports the use of the mature and extensive theory of algebraic graph transformations for graphs with many different and also uncommon hierarchy concepts. The aim of our approach is not a generalisation of hierarchy concepts in graph transformation but a possibility to access algebraic graph transformation for graphs with a wide spectrum of hierarchy concepts.

The vision is a clear and simple access that provides a potential user with the hierarchical technique that is most adequate for the purpose. This requires

a much deeper treatment of the hierarchical concepts at the abstract categorical level as well as an intuitive representation of these concepts.

To aim at this vision future work comprises then formulation of typical results and notions for hierarchical graph transformations, as e.g. flattening, hierarchical rule application or imposing a hierarchy. Moreover, the inclusion of labels, types and attributes is central for realizing that vision. For this task the work in [21] is an exciting prospect. Additionally, the transfer of existing concepts to this more categorical approach is required based on further investigation of the relations discussed in Sect. 5.

Acknowledgements. I am very grateful for the constructive and thorough comments of the anonymous referees.

References

1. Adamek, J.: Introduction to coalgebra. *Theor. Appl. Categories* **14**, 157–199 (2005). <http://www.tac.mta.ca/tac/volumes/14/8/14-08abs.html>
2. AGG: The attributed graph grammar system (2014). <http://user.cs.tu-berlin.de/~gragra/agg/>, revision: 10/29/2014 16:43:00
3. Benford, S., Calder, M., Rodden, T., Sevegnani, M.: On lions, impala, and bigraphs: Modelling interactions in physical/virtual spaces. *ACM Trans. Comput. Hum. Interact.* **23**(2), 9: 1–9: 56 (2016). <http://doi.acm.org/10.1145/2882784>(2016)
4. Bruni, R., Corradini, A., Montanari, U.: Modeling a service and session calculus with hierarchical graph transformation. *ECEASST* 30 (2010). <http://journal.ub.tu-berlin.de/index.php/eceasst/article/view/427>
5. Bruni, R., Montanari, U., Plotkin, G.D., Terreni, D.: On hierarchical graphs: reconciling bigraphs, Gs-monoidal theories and Gs-graphs. *Fundam. Inform.* **134**(3–4), 287–317 (2014). <http://dx.doi.org/10.3233/FI-2014-1103>
6. Busatto, G.: An abstract model of hierarchical graphs and hierarchical graph transformation. Ph.D. thesis, University of Paderborn, Germany (2002). <http://ubdata.uni-paderborn.de/ediss/17/2002/busatto/disserta.pdf>
7. Busatto, G., Hoffmann, B.: Comparing notions of hierarchical graph transformation. *Electr. Notes Theor. Comput. Sci.* **50**(3), 310–317 (2001). [http://dx.doi.org/10.1016/S1571-0661\(04\)00184--7](http://dx.doi.org/10.1016/S1571-0661(04)00184--7)
8. Busatto, G., Kreowski, H., Kuske, S.: Abstract hierarchical graph transformation. *Math. Struct. Comput. Sci.* **15**(4), 773–819 (2005). <http://dx.doi.org/10.1017/S0960129505004846>
9. Drewes, F., Hoffmann, B., Plump, D.: Hierarchical graph transformation. *J. Comput. Syst. Sci.* **64**(2), 249–283 (2002). <http://dx.doi.org/10.1006/jcss.2001.1790>
10. Ehrig, H., Ehrig, K., Prange, U., Taentzer, G.: *Fundamentals of Algebraic Graph Transformation*. EATCS Monographs in TCS. Springer (2006)
11. Ehrig, H.: Bigraphs meet double pushouts. *Bull. ATCS* **78**, 72–85 (2002)
12. Ehrig, H., Golas, U., Hermann, F.: Categorical frameworks for graph transformation and HLR systems based on the DPO approach. *Bull. EATCS* **102**, 111–121 (2010)
13. Engels, G., Schürr, A.: Encapsulated hierarchical graphs, graph types, and meta types. *Electr. Notes Theor. Comput. Sci.* **2**, 101–109 (1995). [http://dx.doi.org/10.1016/S1571-0661\(05\)80186--0](http://dx.doi.org/10.1016/S1571-0661(05)80186--0)

14. Ermel, C., Rudolf, M., Taentzer, G.: The agg approach: language and environment. In: Ehrig, H., Engels, G., Kreowski, H.J., Rozenberg, G. (eds.) *Handbook of Graph Grammars and Computing by Graph Transformation*, pp. 551–603. World Scientific Publishing Co., Inc. (1999). <http://dl.acm.org/citation.cfm?id=328523.328619>
15. Gadducci, F., Heckel, R.: An inductive view of graph transformation. In: Presicce, F.P. (ed.) *WADT 1997. LNCS*, vol. 1376, pp. 223–237. Springer, Heidelberg (1998). doi:[10.1007/3-540-64299-4_36](https://doi.org/10.1007/3-540-64299-4_36)
16. Gassara, A., Rodriguez, I.B., Jmaiel, M., Drira, K.: Encoding bigraphical reactive systems into graph transformation systems. *Electron. Notes Discrete Math.* **55**, 207–210 (2016). <http://dx.doi.org/10.1016/j.endm.2016.10.051>
17. Jäkel, C.: A unified categorical approach to graphs (2015). <https://arxiv.org/abs/1507.06328>
18. Jäkel, C.: A coalgebraic model of graphs (2016). <https://arxiv.org/abs/1508.02169>
19. Junghanns, M., Petermann, A., Rahm, E.: Distributed grouping of property graphs with gradoop. In: *Proceedings of the 17. Fachtagung, Datenbanksysteme für Business, Technologie und Web. LNI, GI* (2017) (to be published)
20. Kahl, W.: Categories of coalgebras with monadic homomorphisms. In: Bonsangue, M.M. (ed.) *CMCS 2014 2014. LNCS*, vol. 8446, pp. 151–167. Springer, Heidelberg (2014). doi:[10.1007/978-3-662-44124-4_9](https://doi.org/10.1007/978-3-662-44124-4_9)
21. Kahl, W.: Graph transformation with symbolic attributes via monadic coalgebra homomorphisms. *ECEASST* 71 (2014). <http://journal.ub.tu-berlin.de/eceasst/article/view/999>
22. Milner, R.: Pure bigraphs: structure and dynamics. *Inf. Comput.* **204**(1), 60–122 (2006). <http://dx.doi.org/10.1016/j.ic.2005.07.003>
23. Padberg, J.: Towards \mathcal{M} -adhesive categories of coalgebraic graphs. Technical report, ArXiv e-prints (2017). <https://arxiv.org/abs/1702.04650>
24. Palacz, W.: Algebraic hierarchical graph transformation. *J. Comput. Syst. Sci.* **68**(3), 497–520 (2004). [http://dx.doi.org/10.1016/S0022-0000\(03\)00064-3](http://dx.doi.org/10.1016/S0022-0000(03)00064-3)
25. Prange, U., Ehrig, H., Lambers, L.: Construction and properties of adhesive and weak adhesive high-level replacement categories. *Appl. Categorical Struct.* **16**(3), 365–388 (2008)
26. Rutten, J.: Universal coalgebra: a theory of systems. *Theor. Comput. Sci.* **249**(1), 3–80 (2000). <http://www.sciencedirect.com/science/article/pii/S0304397500000566>
27. Schneider, H.J.: Describing systems of processes by means of high-level replacement. In: *Handbook of Graph Grammars and Computing by Graph Transformation*, vol. 3, pp. 401–450. World Scientific (1999)
28. Sevegnani, M., Calder, M.: Bigraphs with sharing. *Theor. Comput. Sci.* **577**, 43–73 (2015). <http://dx.doi.org/10.1016/j.tcs.2015.02.011>
29. Ślusarczyk, G., Lachwa, A., Palacz, W., Strug, B., Paszyńska, A., Grabska, E.: An extended hierarchical graph-based building model for design and engineering problems. *Autom. Const.* **74**, 95–102 (2017)
30. Walton, L.A., Worboys, M.: A qualitative bigraph model for indoor space. In: Xiao, N., Kwan, M.-P., Goodchild, M.F., Shekhar, S. (eds.) *GIScience 2012. LNCS*, vol. 7478, pp. 226–240. Springer, Heidelberg (2012). doi:[10.1007/978-3-642-33024-7_17](https://doi.org/10.1007/978-3-642-33024-7_17)
31. Worboys, M.F.: Using bigraphs to model topological graphs embedded in orientable surfaces. *Theor. Comput. Sci.* **484**, 56–69 (2013). <http://dx.doi.org/10.1016/j.tcs.2013.02.018>

Graph Transformation

10th International Conference, ICGT 2017, Held as Part
of STAF 2017, Marburg, Germany, July 18-19, 2017,
Proceedings

de Lara, J.; Plump, D. (Eds.)

2017, XIV, 231 p. 76 illus., Softcover

ISBN: 978-3-319-61469-4