Composition of Transformations: A Framework for Systems with Dynamic Topology *

Marnes Augusto Hoff, Karina Girardi Roggia, Paulo Blauth Menezes
Instituto de Informática, Universidade Federal do Rio Grande do Sul
Caixa Postal 15064, 91501-970, Porto Alegre, Brazil
Phone: +55 (51) 3316-6159 - 3316-6168; Fax: +55 (51) 3316-7308
{marnes,kaqui,blauth}@inf.ufrgs.br

Abstract In graph-based systems there are many methods to compose (possibly different) graphs. However, none of these usual compositions are adequate to naturally express semantics of systems with dynamic topology, i.e., systems whose topology admits successive transformations through its computation. We constructed a categorical semantic domain for graph based systems with dynamic topology using a new way to compose edges of (possible different) graphs. In this context, sequences of different graphs represent successive transformations of system topology during its computation and the edges composition between those graphs, the semantics of the corresponding dynamic system. Then we show how the proposed approach can be used to give semantics to concurrent anticipatory systems.
Keywords : Graphs, Category Theory, Composition, Anticipatory Systems, Dynamical Topology.

1 Introduction

In graph-based systems there are many methods to compose (possibly different) graphs. Some examples are the parallel and nondeterministic compositions usually reflected by the categorical constructions of product and coproduct ([12], [9], [11]), respectively. Also, there are some non-traditional ways to compose systems such as the graph-grammar using double-pushout (total morphisms) ([5], [10]) or single-pushout (partial morphisms) ([7], [9], [6]) approaches. However, none of these compositions are adequate to naturally express semantics of systems with dynamic topology, i.e., systems whose topology admits successives transformations through its computation.

For example, consider the graph-based system in figure 1 (left). Suppose that, after the computation of the transition 'a', the system changes its topology, replacing the arc b by the arcs p, q, r and s as in figure 1 (right). For instance, this kind of dynamical topological change can be done using graph grammar approaches as in

---

[9]. Suppose that, after the graph transformation, the transition 'p' occurs followed by the transition 'r' (so, between these last transitions, the topology of the system didn't change). This behavior can be visualized in figure 2. In figure 3 we can see the semantics of this process; in the left, the semantics after the first two possible steps and, in the right, the semantics of all possible three steps.
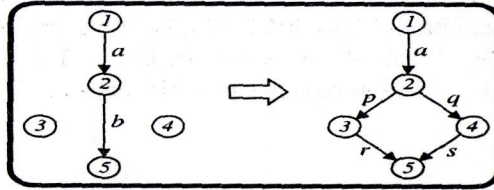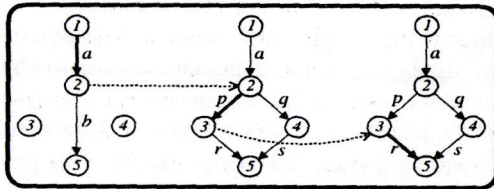

Fig. 1: A Graph Transformation

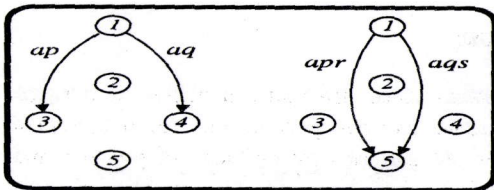**Fig. 2**: Example of a System Computation

Fig. 3: Semantics of the System

This kind of non-traditional way of composing graphs is similar to the mathematical composition of a special type of relation: the partial maps. Indeed, we show that extending the composition of partial maps (Set Theory) to a categorical context, the expected semantics domain for graph-based system with dynamic topology is reached.

The proposed solution satisfies several important properties with relevant computational interpretations specially in the context of concurrent, anticipatory systems ([4], [13]). For instance: (i) it satisfies the vertical and the horizontal compositionality requirements , i.e, every dynamic system has a semantics and the semantics of

concurrent dynamic systems is the parallel composition of semantics of component dynamic systems, respectively; (ii) it is able to give semantics of dynamic steps in any order (even backwards); (iii) it has some basic characteristics of dynamical systems such as the attractor and "no operation" notions.

For simplicity, in this paper, we suppose that the changes of topology of a graph-based system doesn't affect the set of states (nodes).

2 Composition of Edges of Dynamic Systems

Considering that the basic idea is to generalize the composition of partial maps in a categorical context in order to give semantics to systems that evolve (modifying topologies) during their computations, the following approach starts with the categorical notions of partial maps and composition.

Consider the figure 4 (left). A partial map $F : A \rightarrow B$, in Category Theory, can be seen as a span ([3], [2]) $\langle DF, f_A, f_B \rangle$ with $f_A : DF \rightarrow A$ and $f_B : DF \rightarrow B$, where f_A is monic. The object DF of the span together with the mono f_A represents the subobject of A where the partial map F is defined. Now consider the figure 5 (left). In this way, the mathematical binary composition $G \circ F : A \rightarrow C$ of partial maps $F : A \rightarrow B$ and $G : B \rightarrow C$ is defined categorically using a pullback of f_B and g_B. Here, being p_0 and p_1 morphisms of the resulting pullback, the composed partial map $H = G \circ F$ is $\langle DH, h_A. h_C \rangle$ where $h_A = f_A \circ p_0$ and $h_C = g_C \circ p_1$.

In this context, we can see a graph as a generalized partial map without the restriction of one arrow be monic ($T : V \rightarrow V$, where T and V are the sets of arcs and nodes respectively and $\delta_0, \delta_1 : T \rightarrow V$ are the source and target functions, respectively) as in figure 4 (right), and the composition of graphs as the generalized composition of partial maps as in figure 5 (right).

Definition 1 (Graph) *Let V be a set of nodes, i.e., an object in Set. A graph in V is a span $\langle T, \delta_0, \delta_1 \rangle$ with $\delta_0 : T \rightarrow V$ and $\delta_1 : T \rightarrow V$, as in figure 4 (right), where T is a set of edges and δ_0 and δ_1 represent the source and target functions, respectively.*
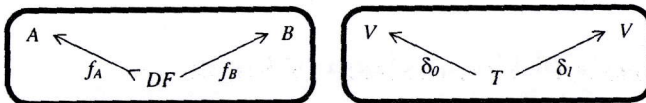

Fig. 4: Partial Map (left) and Graph (right)

Definition 2 (Partial Map) *Let A and B be sets. A partial map $F : A \rightarrow B$ is a span $\langle DF, f_A, f_B \rangle$ with $f_A : DF \rightarrow A$ and $f_B : DF \rightarrow B$, with the restriction that f_A is monic, like in figure 4 (left).*
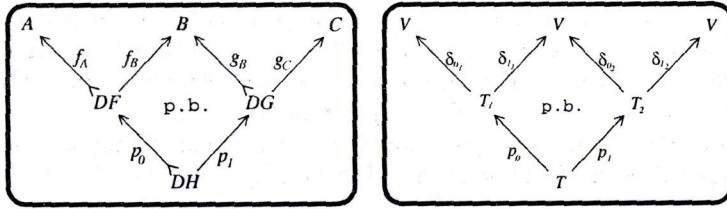
Fig. 5: Composition of Partial Maps (left) and Composition of Graphs (right)

Note that each endorrelation $(R : A \to A)$ is a special case of graphs where the unique arrow $R \to A \times A$ is monic.

Now we define the (binary) composition of edges of (possible different) graphs, which we interpret as the edges composition of dynamic systems. Remember that, for simplicity in this paper, the graphs have the same set of nodes. We denote the composed graph $G_2 \circ G_1$ by $G_1 \triangleright G_2$.

Definition 3 (Composition of Edges) *Let V be a set of nodes and $G_1 = \langle V, T_1, \delta_{0_1}, \delta_{1_1} \rangle$ and $G_2 = \langle V, T_2, \delta_{0_2}, \delta_{1_2} \rangle$ be graphs in V. The Binary Composition of Edges of Graphs, Composition of Edges for short, of G_1 and G_2 is the graph $G_1 \triangleright G_2 = \langle V, T, \delta_0, \delta_1 \rangle$ where T is the resulting object of the pullback shown in figure 5 (right), $\delta_0 = \delta_{0_1} \circ p_0$ and $\delta_1 = \delta_{1_2} \circ p_1$.*

Each edge in the resulting graph $G_1 \triangleright G_2$ represents a path of length 2 (between nodes), which first half is some edge of graph G_1 and which second half is some edge of graph G_2. For example, see figure 3 (left).

In a graph-based system, the composed edge of $G_1 \triangleright G_2$ can be seen as a computation resulting of sequential composition of another two: the first occurs in system G_1 and the second, in system G_2 (in this order). Yet, it may be interpreted as a computation over a system with context switch in the sense of [14], where the first computation occurs before and the second occurs after the switching. Note that the context switch may be interpreted as a transformation, a mutation, an evolution or a reorganization of the topology of a system.

3 Properties of Composition of Edges

The composition of edges satisfies several important properties with relevant computational interpretations specially in the context of anticipatory systems. The following properties are proved:

• closure: satisfies the vertical compositionality requirement, i.e., every dynamic system has semantics;

• associativity: the semantics of the dynamic steps of a system may be calculated in any order, even backwards;

- absorbent element: reflects an attractor notion;
- identity element: reflects the "no operation" notion if we see the arcs of the identity graph as NOP transitions. This interpretation is very clear in a category of reflexive graphs which is not the scope of this paper;
- non-commutativity: since systems are able to modify their topologies, the order of transformations matters;
- non-idempotency, together with the associativity property, induces the n-ary operation of composition of edges;
- properties for concurrent dynamic systems: while the composition of edges doesn't distribute through parallel compositional (categorical product) it satisfies a very important and non-usual requirement which is the horizontal compositionality (interchange law in the sense of [8]), i.e., the semantics of concurrent dynamic systems is the parallel composition of semantics of component dynamic systems.

Property 1 (Closure) *Let* V *be a set of nodes and* $G_1 = \langle V, T_1, \delta_{0_1}, \delta_{1_1} \rangle$ *and* $G_2 = \langle V, T_2, \delta_{0_2}, \delta_{1_2} \rangle$ *be graphs in* V. *So the composition* $G_1 \triangleright G_2 = \langle V, T, \delta_0, \delta_1 \rangle$ *is also a graph in* V.

Proof. Since *Set* is complete, the composition $G_1 \triangleright G_2 = \langle V, T, \delta_0, \delta_1 \rangle$ always exists. Additionally, it is obviously a graph in V, due both δ_0 and δ_1 are arrows $T \to V$.

Property 2 (Associativity) *Let* V *be a set of nodes and* $G_1 = \langle V, T_1, \delta_{0_1}, \delta_{1_1} \rangle$, $G_2 = \langle V, T_2, \delta_{0_2}, \delta_{1_2} \rangle$ *and* $G_3 = \langle V, T_3, \delta_{0_3}, \delta_{1_3} \rangle$ *be any graphs in* V, *so* $(G_1 \triangleright G_2) \triangleright G_3 = G_1 \triangleright (G_2 \triangleright G_3)$.

Proof. Figure 6 shows both $(G_1 \triangleright G_2) \triangleright G_3$ (top) and $G_1 \triangleright (G_2 \triangleright G_3)$ (bottom). To construct $(G_1 \triangleright G_2) \triangleright G_3$, one calculates the pullback of δ_{1_1} and δ_{0_2} which is $\langle P_{12}, f_{A_{12}}, f_{B_{12}} \rangle$ and so the pullback of $\delta_{1_2} \circ f_{B_{12}}$ and δ_{0_3} which is $\langle P_{(12)3}, f_{A_{(12)3}}, f_{B_{(12)3}} \rangle$. To construct $G_1 \triangleright (G_2 \triangleright G_3)$, one calculates the pullback of δ_{1_2} and δ_{0_3} which is $\langle P_{23}, f_{A_{23}}, f_{B_{23}} \rangle$ and so the pullback of $\delta_{0_2} \circ f_{A_{23}}$ and δ_{1_1} which is $\langle P_{1(23)}, f_{A_{1(23)}}, f_{B_{1(23)}} \rangle$.

To show $P_{(12)3}$ and $P_{1(23)}$ are isomorphic, we need first to find two morphism, one $P_{(12)3} \to P_{1(23)}$ and other $P_{1(23)} \to P_{(12)3}$, and then we show they are iso. Since $\langle P_{23}, f_{A_{23}}, f_{B_{23}} \rangle$ is the pullback of δ_{1_2} and δ_{0_3}, then $\langle P_{(12)3}, f_{B_{12}} \circ f_{A_{(12)3}}, f_{B_{(12)3}} \rangle$ is a pre-pullback of δ_{1_2} and δ_{0_3}, so there is an unique arrow $1 : P_{(12)3} \to P_{23}$ that commutes, as shown in figure 7 (left). Since $\langle P_{1(23)}, f_{A_{1(23)}}, f_{B_{1(23)}} \rangle$ is the pullback of δ_{1_1} and $\delta_{0_2} \circ f_{A_{23}}$, then $\langle P_{(12)3}, f_{A_{12}} \circ f_{A_{(12)3}}, 1 \rangle$ is a pre-pullback of δ_{1_1} and $\delta_{0_2} \circ f_{A_{23}}$, so there is an unique arrow $2 : P_{(12)3} \to P_{1(23)}$ that commutes, as shown in figure 7 (center). Analogously, we can find a unique arrow $4 : P_{1(23)} \to P_{(12)3}$ that commutes.

Finally, since $\langle P_{1(23)}, f_{A_{1(23)}}, f_{B_{1(23)}} \rangle$ is the pullback of δ_{1_1} and $\delta_{0_2} \circ f_{A_{23}}$, then $\langle P_{1(23)}, f_{A_{1(23)}} \circ 2 \circ 4, f_{B_{1(23)}} \circ 2 \circ 4 \rangle$ is a pre-pullback of δ_{1_1} and $\delta_{0_2} \circ f_{A_{23}}$, so there is an unique arrow $P_{1(23)} \to P_{1(23)}$ that commutes, as shown in figure 7 (right). It is obvious that this arrow is $id_{P_{1(23)}}$ and $2 \circ 4 = id_{P_{1(23)}}$. In the same way, we show $4 \circ 2 = id_{P_{(12)3}}$.

So, $P_{(12)3}$ and $P_{1(23)}$ are isomorphic and the associativity of $\triangleright$ holds. We can now freely write $G_1 \triangleright G_2 \triangleright G_3$, omitting the parenthesis without any loss of preciseness.
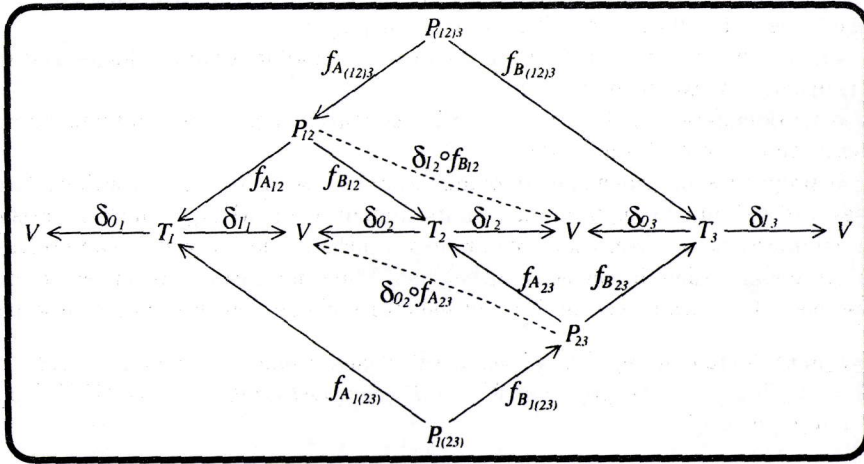

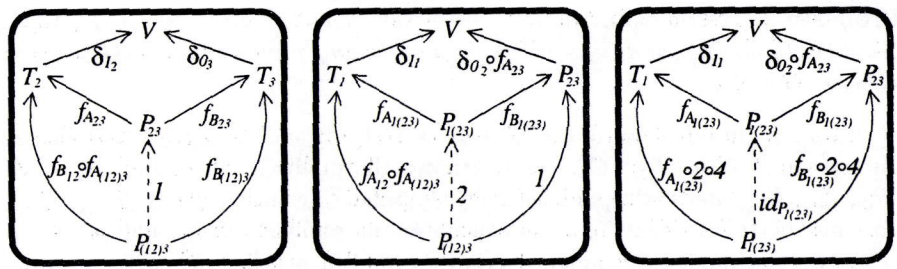**Fig. 6**: Associativity Diagram

Fig. 7: Commutative Diagrams to Associativity

Each edge in the resulting graph $G_1 \triangleright G_2 \triangleright G_3$ represents a path of length 3 (between nodes), whose first third is some edge of graph G_1, whose second third is some edge of graph G_2 and whose last third is some edge of graph G_3. In fact, to any n graphs $G_1, G_2, ..., G_n$ in a set V of nodes, each edge in $G_1 \triangleright G_2 \triangleright ... \triangleright G_n$ represents a path of length n whose ith nth-part belongs to G_i, where $1 \leq i \leq n$. This can easily be proved by induction.

As a consequence of associativity, the computations through compositions in graph-based systems can be calculated in any order, even backwards. Therefore, this property allow us to optimize the calculus of an n-ary composition in a parallel way.

Property 3 (Absorbent Element) *To any set V of nodes, there is a graph $Z_V = \langle V, \varnothing, \varnothing, \varnothing \rangle$ which is the absorbent element for the composition $\triangleright$ of graphs in V.*

Proof. To see that Z_V is a left absorbent, let V be a set of nodes and $T_2 = \langle V, T_2, \delta_{0_2}, \delta_{1_2} \rangle$ be any graph in V. Seeing figure 8 (right), becomes obvious that P is $\varnothing$. Idem to f_A and f_B. The resulting graph is $Z_V \triangleright T_2 = \langle V, \varnothing, \varnothing \circ \varnothing, \delta_{1_2} \circ \varnothing \rangle = \langle V, \varnothing, \varnothing, \varnothing \rangle = Z_V$. The proof that Z_V is also a right absorbent is analogous.

In semantics of systems, an absorbent element works as an attractor.

Property 4 (Identity Element) *To any set V of nodes, there is a graph $I_V = \langle V, V, id_V, id_V \rangle$ which is the identity element for composition $\triangleright$ of graphs in V.*

Proof. To see that I_V is a left identity, let V be a set of nodes and $T_2 = \langle V, T_2, \delta_{0_2}, \delta_{1_2} \rangle$ be any graph in V. As we can see in figure 8 (left), f_B is iso by preserving the pullback opposite arrow properties, so P and f_B may be respectively T_2 and id_{T_2}. Now, f_A becomes δ_{0_2}. The resulting graph is $I_V \triangleright T_2 = \langle V, T_2, id_V \circ \delta_{0_2}, \delta_{1_2} \circ id_{T_2} \rangle = \langle V, T_2, \delta_{0_2}, \delta_{1_2} \rangle = T_2$. The proof that I_V is also a right identity is analogous.
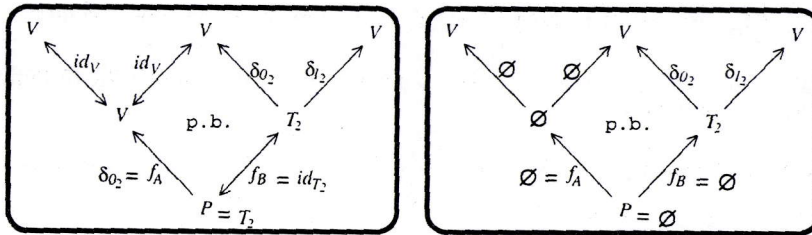

Fig. 8: Left Identity Element (left) and Left Absorbent Element (right)

Property 5 (Non-commutativity) *Let V be a set of nodes and $G_1 = \langle V, T_1, \delta_{0_1}, \delta_{1_1} \rangle$ and $G_2 = \langle V, T_2, \delta_{0_2}, \delta_{1_2} \rangle$ be graphs in V. So, generally, $G_1 \triangleright G_2 \neq G_2 \triangleright G_1$.*

Proof. For instance, let G_1 and G_2 be the graphs 1 and 2 in figure 9, respectively. It's easy to see that $G_1 \triangleright G_2 \neq G_2 \triangleright G_1$
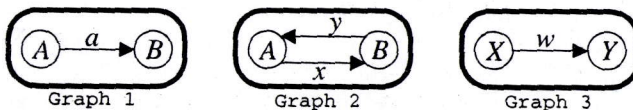

Fig. 9: Example graphs

265

Property 6 (Non-idempotency) *Let V be a set of nodes and $G = \langle V, T, \delta_0, \delta_1 \rangle$ be a graph in V. So, generally, $G \triangleright G \neq G$.*

Proof. For instance, let G be the graph 2 in figure 9. It's easy to see that $G \triangleright G \neq G$.

The fact this operator does not satisfy idempotency carry us to the study of successive compositions of the same graph, i.e., $G \triangleright G$, $G \triangleright G \triangleright G$, and so on, denoted by G^n. First note that $G^1 = G$ and $G^0 = I_V$: (i) to see that $G^1 = G$ consider the diagram inside the doted lines in figure 10 (left) given by the graph $G = \langle V, T, \delta_0, \delta_1 \rangle$ and the corresponding limit, which is G; (ii) to see that $G^0 = I_V$ consider the diagram inside the doted lines in figure 10 (right) given by no graphs in V and the corresponding limit, which is I_V.
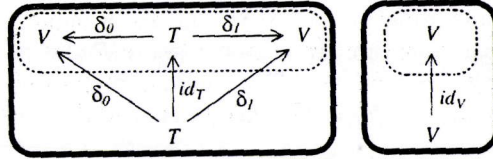

Fig. 10: Limit to a single graph (left) and to "no graphs" (right)

Since this operation is associative, we can make use of induction to define the general case, as follows.

Definition 4 (Inductive Definition to G^n) *Let V be a set of nodes and $G = \langle V, T, \delta_1, \delta_1 \rangle$ be a graph in V. So $G^n = \begin{cases} I_V, & \text{if } n = 0; \\ G \triangleright G^{n-1}, & \text{if } n > 0. \end{cases}$*

In the following text, the graphs categorial product operator is denoted by $\times$. Remember that the graphs categorial product is defined to graphs with (possible) different set of nodes.

Property 7 (Non-distributivity of $\times$ over $\triangleright$) *Let V_{12} and V_3 be sets of nodes, $G_1 = \langle V_{12}, T_1, \delta_{0_1}, \delta_{1_1} \rangle$ and $G_2 = \langle V_{12}, T_2, \delta_{0_2}, \delta_{1_2} \rangle$ be graphs in V_{12} and $G_3 = \langle V_3, T_3, \delta_{0_3}, \delta_{1_3} \rangle$ be a graph in V_3. So, generally, $(G_1 \triangleright G_2) \times G_3 \neq (G_1 \times G_3) \triangleright (G_2 \times G_3)$ (non-left-distributivity).*

Proof. Let G_1, G_2 and G_3 be, respectively, the graphs 1, 2 and 3 in figure 9. It is easy to see that $(G_1 \triangleright G_2) \times G_3 \neq (G_1 \times G_3) \triangleright (G_2 \times G_3)$. Note that, since the $\times$ is commutative, left and right distributivity becomes isomorphic. Also the distributivity of $\triangleright$ over $\times$ is not in the scope of this paper because, generally, it involves the operator $\triangleright$ between graphs with different set of nodes.

Property 8 (Interchange Law with Categorial Product of Graphs) *Let* V_{AB} *and* V_{CD} *be sets of nodes,* $G_A = \langle V_{AB}, T_A, \delta_{0_A}, \delta_{1_A} \rangle$ *and* $G_B = \langle V_{AB}, T_B, \delta_{0_B}, \delta_{1_B} \rangle$ *be graphs in* V_{AB} *and* $G_C = \langle V_{CD}, T_C, \delta_{0_C}, \delta_{1_C} \rangle$ *and* $G_D = \langle V_{CD}, T_D, \delta_{0_D}, \delta_{1_D} \rangle$ *be graphs in* V_{CD}. *So,* $(G_A \triangleright G_B) \times (G_C \triangleright G_D) = (G_A \times G_C) \triangleright (G_B \times G_D)$.

Proof. The proof of the above property is sketched in figure 11 followed by a construction of an isomorphism between $T_{(A \times C) \triangleright (B \times D)}$ and $T_{(A \triangleright B) \times (C \triangleright D)}$.
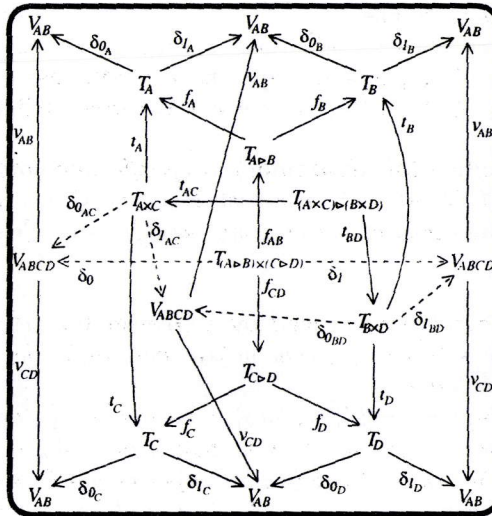

Fig. 11: Interchange Law Diagram

Considering this properties the following important conclusions can be stated (but aren't discussed in this paper):

Corollary 1 ($Gr(V)$ **induces a monoid**): Let V be a set of nodes and $Gr(V) = \{G \mid G \text{ is a graph in } V\}$. So, the algebra $Mon_{Gr(V)} = \langle Gr(V), \triangleright, I_V \rangle$, is a monoid induced by V, since the operator $\triangleright$ is closed in $Gr(V)$, is associative and satisfies left and right identity. There is only one finite induced monoid which is $Mon_{Gr(\varnothing)}$.

Corollary 2 ($Mon_{Gr(V)}$ **is category**): Let V be a set of nodes. So, $Mon_{Gr(V)}$ is a category with a single object.

Corollary 3 ($Mon_{Gr_{Set}}$ **is subcategory of** Mon): To any set V of nodes, $Mon_{Gr(V)}$ is an object of the Mon (category of monoids). Let $Mon_{Gr_{Set}}$ be the category of all monoids induced by objects of the Set category. So, $Mon_{Gr_{Set}}$ is a subcategory of Mon.

4 Composition of Transformations: Giving Semantics to Anticipatory Systems

To illustrate how the composition of edges between (possible different) graphs can give semantics to anticipatory systems, consider the previous work based on Petri Nets (viewed as graphs) transformations [9] where "a specification grammar can be viewed as a specification of a system and the induced subcategory as all possible dynamic anticipations of the system (objects) and their relationship (morphism)", restricting Petri Nets to graphs.

In this context, the following definitions consider the category $p\mathcal{G}r$ (as in [1], i.e., the partial category of graphs and graphs homomorphisms) instead of $p\mathcal{MP}etri$ (the partial category of Marked Petri Nets and its corresponding morphisms).

Definition 5 (Specification Grammar) *A specification grammar or just grammar is Gram $= \langle R, I, N \rangle$ where R, I are collections of $p\mathcal{G}r$-morphisms representing the rules and instantiations of the grammar and N is a $p\mathcal{G}r$-object called initial graph.*

Definition 6 (Subcategory Induced by a Grammar) *Let Gram $= \langle R, I, N \rangle$ be a grammar. The subcategory $\mathcal{G}ram$ of $p\mathcal{G}r$ induced by the grammar Gram is inductively defined as follows:*

a) N is an $\mathcal{G}ram$-object and $[\langle id_N, id_N \rangle] : N \to N$ is a $\mathcal{G}ram$-morphism;

b) for all $\mathcal{G}ram$-object M, for all instantiation $m_0 : M_0 \to M$ and for all rule $r : M_0 \to P_0$, $[\langle r, m_0 \rangle] : M \to P$ is a $\mathcal{G}ram$-morphism and P is an $\mathcal{G}ram$-object;

c) for all $\mathcal{G}ram$-morphisms $\varphi : M \to P$, $\psi : P \to Q$, the morphism $[\langle \psi \circ \varphi, id_M \rangle] : M \to Q$ is a $\mathcal{G}ram$-morphism.

Therefore, for an initial graph N and a grammar $Gram = \langle R, I, N \rangle$, the induced subcategory corresponds to a tree with all possible anticipations as in figure 12. Note that, for each branch in the tree (sequential steps of graph transformations) the defined edge composition of component graphs gives the branch semantics. Thus, calculating the compositions of all branches we have the semantics of all possible dynamic anticipations of the given system.
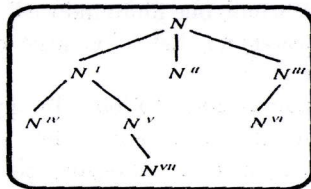

Fig. 12: Tree with all Possible Anticipations

Briefly, if the three graphs in figure 2 represent a branch in a tree with two steps of transformations, then the graph in figure 3 (right) gives the corresponding semantics.

5   Conclusions and Future Works

We constructed a categorical semantic domain for graph based systems with dynamic topology using a new way to compose edges of (possible different) graphs. In this context, sequences of different graphs represent successive transformations of system topology during its computation and the edges composition between those graphs, the semantics of the corresponding dynamic system.

The composition of edges between graphs is inspired in the mathematical composition of partial maps generalized for graphs using the categorical pullback construction resulting in a semantic domain that is powerful but simple and precisely defined.

The proposed mechanism satisfies several important properties with relevant computational interpretations specially in the context of concurrent anticipatory systems such as the diagonal compositionality requirement, i.e., every dynamic system has a semantics (horizontal) and the semantics of concurrent dynamic systems is the parallel composition of semantics of component dynamic systems (vertical). In fact, we show that the composition of edges is able to give semantics to anticipatory systems.

Exploiting the categorical constructions over the proposed composition of edges, a full calculus of dynamic systems should arise. Moreover, following an approach similar to [10], generalizations for several graph based systems, such as (dynamic) transition systems and (dynamic) Petri Nets, can be reached. Also we are working on a notion of equivalence of systems with dynamical topology.

Some other works are: an investigation of colimits properties, a framework for reflexive graphs (possible giving a notion of computational closure) and an investigation of conditions for fixed point.

References

[1] Asperti, Andrea; Longo, Giuseppe (1991). *Categories, Types, and Structures - An Introduction to Category Theory for the Working Computer Scientist.* MIT Press.

[2] Barbosa, Luís Soares (2003). A brief introduction to bicategories. Techn. Report DI-PURe-03:12:01, Departamento de Informática da Universidade do Minho.

[3] Bénabou, J. (1967) Introduction to bicategories. In *Reports of the Midwest Category Seminar*, in Springer Lecture Notes in Mathematics, Springer-Verlag. number 47, pages 1–77.

[4] Dubois, D. M. (1998) Introduction to Computing Anticipatory Systems. In *International Journal of Computing Anticipatory Systems*, CHAOS. volume 2, pages 3–23.

[5] Ehrig, Hartmut (1979). Introduction to the algebraic theory of graph grammars. In V. Claus, H. Ehrig, and G. Rozenberg, editors, *Graph-Grammars and Their Application to Computer Science and Biology*, volume 73 of *Lecture Notes in Computer Science*, pages 1–69.

[6] Kennaway, Richard (1991). Graph Rewriting in Some Categories of Partial Morphisms. In Hartmut Ehrig, Hans-Jörg Kreowski, and Grzegorz Rozenberg, editors, *Proc. 4th. Int. Workshop on Graph Grammars and their Application to Computer Science*, Springer-Verlag. volume 532 of *Lecture Notes in Computer Science*, pages 490–504.

[7] Löwe, Michael (1993). Algebraic approach to single-pushout graph transformation. *Theoretical Computer Science*, 109(1–2):181–224.

[8] Mac Lane, Saunders (1997). *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics*. Springer-Verlag, 2nd edition.

[9] Menezes, Paulo Blauth (1999). A categorical framework for concurrent anticipatory systems. In *Proceedings of 2nd International Conference on Computing Anticipatory Systems*, American Institute of Physics. *AIP Conference Proceedings* 465. pages 185–199.

[10] Menezes, Paulo Blauth (2000). Duo-Internal Labeled Graphs with Distinguished Nodes: a categorial framework for graph based anticipatory systems. *International Journal Of Computing Anticipatory Systems*, 6:75–93.

[11] Menezes, Paulo Blauth; Costa, Simone A.; Machado, Júlio H. A. P.; Ramos, J. (2002) Nautilus: a concurrent anticipatory programming language. In *Proceedings of 5th International Conference on Computing Anticipatory Systems*, American Institute of Physics. *AIP Conference Proceedings* 627, pages 553–564.

[12] Meseguer, José; Montanari, Ugo (1990). Petri nets are monoids. *Information and Computation*, 88(2):105–155.

[13] Rosen. R. (1985) *Anticipatory Systems*. Pergamon Press.

[14] Tanenbaum, Andrew S (1992). *Modern Operating Systems*. Prentice Hall.