DPO Rewriting and Abstract Semantics via Opfibrations

R. Banach [1]

Computer Science Department,
Manchester University, Manchester, U.K.

Abstract

The classical DPO graph rewriting construction is re-expressed using the opfibration approach introduced originally for term graph rewriting. Using a skeleton category of graphs, a base of canonical graphs-in-context, with DPO rules as arrows, and with categories of redexes over each object in the base, yields a category of rewrites via the discrete Grothendieck construction. The various possible ways of combining rules and rewrites leads to a variety of functors amongst the various categories formed. Categories whose arrows are rewriting sequences have counterparts where the arrows are elementary event structures, and an event structure semantics for arbitrary graph grammars emerges naturally.

Key Words: Graph grammar, DPO rewriting, opfibration, event semantics, event structure.

1 Introduction

Since the opfibration construction for describing (a species of) term graph rewriting was introduced in [1,2], the author has wondered whether something similar would work for the classical double pushout (DPO) rewriting approach [10,11]. (It is clear that the single pushout approach can be accomodated in an opfibration construction.) Taking as objects in the base, arrows $l : K \to L$, allows the Grothendieck construction to be achieved smoothly. This is done in section 2. In section 3 we consider various categories of rewriting sequences and the functors between them. Section 4 shows that the arrows in these rewriting categories have a natural correspondence with event structures, and section 5 treats the preceding considerations with regard to a particular graph grammar. Section 6 contains a discussion.

[1] Email: banach@cs.man.ac.uk

2 The DPO Approach as Opfibration

We omit definition of the category $\mathcal{CG}$ of concrete graphs and *injective* morphisms, and of DPO rewriting.

Let us agree on some canonical construction for the "true" pushout used in the right hand square of the rewrite. It will not matter that we might choose independent canonical constructions for pushout complement and pushout. With such choices, rewriting becomes deterministic, a key ingredient for making an opfibration-based rewriting construction.

Consider now the category $\mathcal{CP}$ (of "concrete patterns") where objects are arrows $l : K \to L$ of $\mathcal{CG}$ and whose arrows are just coinitial ordered pairs of objects, which we can write as $(L \leftarrow K : l, r : K \to R)$ or just (l, r) where no confusion arises, and where identities are $(L \leftarrow K : l, l : K \to L)$, and composition of $(L_1 \leftarrow K : l_1, r_1 : K \to R_1)$ and $(L_2 \leftarrow K : l_2, r_2 : K \to R_2)$ is just $(L_1 \leftarrow K : l_1, r_3 : K \to R_3)$.

Above each $l : K \to L$ in $\mathcal{CP}$ we erect the discrete category $\mathcal{CR}^l$ (of "concrete redexes of l") whose objects are arrows $g : L \to G$ of $\mathcal{CG}$ such that in $g \circ l : K \to L \to G$, g and l satisfy the application condition (DANGL).

Theorem 2.1 *Let (l, r) be an arrow of $\mathcal{CP}$. Then there is a functor $\mathcal{CR}ew^{(l,r)} : \mathcal{CR}^l \to \mathcal{CR}^r$ whose action on redexes is to send $g \circ l : K \to L \to G$ to $h \circ r : K \to R \to H$ where the latter is the result of a direct derivation using (l, r).*

The next step is to attempt to glue the concrete rewriting functors together to get a functor from $\mathcal{CP} \to \mathcal{C}at$. Unfortunately this does not work properly as $\mathcal{CR}ew^{(l,l)}$ is not the identity functor on $\mathcal{CR}^l$ and neither is $\mathcal{CR}ew^{(l_2,l_3)} \circ \mathcal{CR}ew^{(l_1,l_2)}$ equal to $\mathcal{CR}ew^{(l_1,l_3)}$ due to the explicit nature of the rewriting construction. Nevertheless there are canonical isomorphisms from

$$\mathcal{CR}ew^{(l,l)}(g : L \to G) \quad \text{to} \quad (g : L \to G)$$

and from

$$\mathcal{CR}ew^{(l_2,l_3)} \circ \mathcal{CR}ew^{(l_1,l_2)}(g : L \to G) \quad \text{to} \quad \mathcal{CR}ew^{(l_1,l_3)}(g : L \to G).$$

These we can exploit in moving away from concrete rewriting, and going to a more abstract formulation. Specifically we can choose a skeleton category of $\mathcal{CG}$ in which these canonically isomorphic pairs of concrete redexes are represented by the same skeleton redex.

Thus we have a skeleton category of graphs and injective morphisms $\mathcal{SG}$. ¿From this we build the pattern category $\mathcal{P}$ of canonical arrows and coinitial pairs which we write as $\langle l, r \rangle$. Above each $l : K \to L$ in $\mathcal{P}$ we erect the discrete category of canonical redexes $\mathcal{R}^l$ for which the objects are $g : L \to G$ satisfying the application condition and where G is a skeleton graph. Note that when we have several potential canonical arrows (eg. for a $g : L \to G$, because of automorphisms of G), we just choose one of them, once and for all.

Theorem 2.1 now translates to a functor

$$\mathcal{R}ew^{\langle l,r \rangle} : \mathcal{R}^l \to \mathcal{R}^r$$

In addition, these functors now compose together nicely and we have:

Theorem 2.2 *There is a functor $\mathcal{R}ew : \mathcal{P} \to \mathcal{C}at$ such that*

$$\mathcal{R}ew(l) = \mathcal{R}^l$$
$$\mathcal{R}ew(\langle l, r\rangle) = \mathcal{R}ew^{\langle l,r\rangle} : \mathcal{R}^l \to \mathcal{R}^r$$

Thus $\mathcal{R}ew$ maps an object of $\mathcal{P}$, $l : K \to L$, to its category of canonical redexes like $g : L \to G$ satisfying the application condition; and $\mathcal{R}ew$ maps an arrow of $\mathcal{P}$, i.e. a double pushout rule $(L \leftarrow K : l, r : K \to R)$ to the skeletonised rewriting construction that turns objects $g : L \to G$ in $\mathcal{R}^l$ into the corresponding rewritten objects $h : R \to H$ in $\mathcal{R}^r$.

Given the above, we can apply the discrete Grothendieck construction to the functor $\mathcal{R}ew$ to obtain the Grothendieck category $\mathcal{G}(\mathcal{P}, \mathcal{R}ew)$ where objects are pairs formed from an object of the base $\mathcal{P}$, $l : K \to L$ say, and an object in the category above it $\mathcal{R}^l$, $g : L \to G$ say. We will write these pairs as $\langle g \circ l : K \to L \to G\rangle$. The arrows of $\mathcal{G}(\mathcal{P}, \mathcal{R}ew)$ are canonical rewrites; thus

$$\langle l, r\rangle : \langle g \circ l : K \to L \to G\rangle \to \langle h \circ r : K \to R \to H\rangle$$

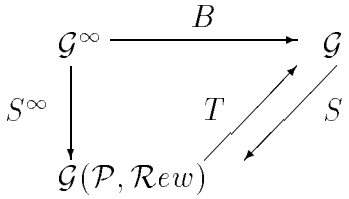
is an arrow iff the $\mathcal{P}$ arrow (rewrite rule), transforms $g \circ l$ into $h \circ r$ in the expected way. The projection functor $F : \mathcal{G}(\mathcal{P}, \mathcal{R}ew) \to \mathcal{P}$ given by

$$F(\langle g \circ l : K \to L \to G\rangle) = (l : K \to L)$$
$$F(\langle l, r\rangle : \langle g \circ l : K \to L \to G\rangle \to \langle h \circ r : K \to R \to H\rangle)$$
$$= \langle l, r\rangle = (L \leftarrow K : l, r : K \to R)$$

is a split opfibration (see [3]).

3 Rewriting Categories

We construct the categories and functors depicted below.



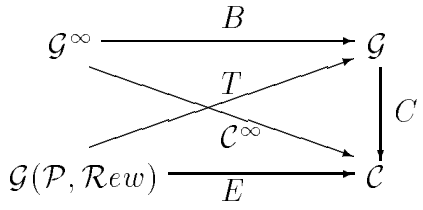
$\mathcal{G}^\infty$ is the free category on the graph built from $\mathcal{G}(\mathcal{P}, \mathcal{R}ew)$ by the equivalence

$$\langle g \circ l : K \to L \to G\rangle \approx \langle h \circ r : K \to R \to H\rangle \quad \text{iff} \quad G = H$$

whose objects are graphs and whose arrows are rewriting sequences. B does (and only does) all internal compositions of rewrites in arrows of $\mathcal{G}^\infty$. Internal compositions are those available according to the law of composition of $\mathcal{G}(\mathcal{P}, \mathcal{R}ew)$, and so $\mathcal{G}$ is a category similar to $\mathcal{G}^\infty$, but no adjacent elements of an arrow are composable in $\mathcal{G}(\mathcal{P}, \mathcal{R}ew)$. T takes arrows of $\mathcal{G}(\mathcal{P}, \mathcal{R}ew)$ to arrows of $\mathcal{G}$ of length 1. S^∞ and S act by

$$S^{(\infty)}([\varepsilon_n; \ldots; \varepsilon_1] : G \to H) =$$
$$\langle \emptyset, \emptyset\rangle : \langle \mathrm{id}_G \circ \emptyset : \emptyset \to G \to G\rangle \to \langle \mathrm{id}_H \circ \emptyset : \emptyset \to H \to H\rangle.$$

We also construct the contents of the diagram below where $\mathcal{C}$ is a category whose arrows are single rewrites under R-composition (which is essentially the R-related composition of [10]), and C, C^∞, E do R-composition on the rewrites in arrows in their domain categories.



Note that we cannot "unify" this diagram with the preceding one because $E \circ S \neq C$ in general. We may regard the category $\mathcal{C}$ as giving a result semantics for rewriting sequences of $\mathcal{G}^\infty$ or $\mathcal{G}$, as in a single arrow, we get the result of all the rewrites. Ironically, if a rewriting sequence $[\varepsilon_n; \ldots; \varepsilon_1] : G_0 \to G_n$ transforms G_0 sufficiently drastically, so that nothing is left of G_0 at the end, then $\varepsilon_n \circ \ldots \circ \varepsilon_1$ in $\mathcal{C}$ will agree with the arrow $E \circ S([\varepsilon_n; \ldots; \varepsilon_1])$ after all.

4 Event Semantics

In this section, we look at the consequences of sequential independence for rewriting sequences in the categorical framework we have set up. We consider $\mathcal{G}^\infty$ first.

Lemma 4.1 *Let $[\varepsilon_1'; \varepsilon_2'] : G_0 \to G_2$ be obtained from $[\varepsilon_2; \varepsilon_1] : G_0 \to G_2$ by swapping sequentially independent rewrites. Then (with rules $\langle l_0, r_1\rangle, \langle l_1, r_2\rangle$), $[\varepsilon_1'; \varepsilon_2'] = [\varepsilon_2; \varepsilon_1]$ iff $l_0 = l_1$ and $r_1 = r_2$.*

Definition 4.2 Let $[\varepsilon_2; \varepsilon_1] : G_0 \to G_2$ be an arrow of $\mathcal{G}^\infty$. If $[\varepsilon_1'; \varepsilon_2'] : G_0 \to G_2$ is obtained from $[\varepsilon_2; \varepsilon_1] : G_0 \to G_2$ by swapping sequentially independent rewrites, and $[\varepsilon_1'; \varepsilon_2'] \neq [\varepsilon_2; \varepsilon_1]$, then we say that ε_1 and ε_2 are independent. Otherwise they are dependent. Similarly if ε_1 and ε_2 are adjacent rewrites in a longer arrow of $\mathcal{G}^\infty$.

Note that our definition forces the following situations to be regarded as dependent: (a), two rewrites by an autoconcurrent non-consumptive rule that does not damage its redex, (thus in a concrete rewriting model, allowing both rewrites of the same redex to be done simultaneously); (b), two rewrites where in a concrete rewriting model, there are two disjoint concrete redexes for the same rule, differing by an automorphism of the rewritten graph; (c), two rewrites by the same rule where the RHS of the rule creates a fresh redex for itself, (i.e. where the rule supports a non-trivial morphism $y : L \to R$). In some of these cases, one would prefer to regard the situation as concurrent, in others not. But due to our using a deterministic rewriting model with canonical redexes, all must be deemed dependent, otherwise a rewriting event could become the same as a predecessor or successor, and the identity of individual events would be lost.

Definition 4.3 Let $\theta = [\varepsilon_n; \ldots; \varepsilon_1] : G_0 \to G_n$ be an arrow of $\mathcal{G}^\infty$ and let Θ be the set of arrows of $\mathcal{G}^\infty$ generated by permuting independent rewrites in θ, and in arrows generated thereby. An event on Θ is a function $e : \Theta \to [1 \ldots n]$ such that if θ and ψ are related by swapping independent $\varepsilon_{i+1}; \varepsilon_i$ in θ, and

$e(\theta) = i+1$ and $f(\theta) = i$, then $e(\psi) = i$ and $f(\psi) = i+1$, and $c(\theta) = c(\psi)$ for all other events c on Θ. (Clearly there are exactly n events on Θ.) Let the set of events on Θ be E. We write $e \leq f$ iff $e(\phi) \leq f(\phi)$ for all $\phi \in \Theta$.

Let $\mathrm{E} = (E, \leq)$. Then E is an elementary event structure. In this way we get the functor

$$M^\infty : \mathcal{G}^\infty \to \mathcal{T}^\infty$$

Here, $\mathcal{T}^\infty$ has as objects graphs G, and as arrows elementary event structures.

The situation with the category $\mathcal{G}$ is a little more interesting. Suppose $[\varepsilon_3; \varepsilon_2; \varepsilon_1] : G_0 \to G_3$ is an arrow of $\mathcal{G}$ and that $\varepsilon_2; \varepsilon_1$ are independent. Swapping them, getting say $[\varepsilon_3; \varepsilon_1'; \varepsilon_2']$, we may find that $\varepsilon_3; \varepsilon_1'$ are internally composable, so that the swap in reality produces a shorter arrow $[\varepsilon_4; \varepsilon_2']$. In such a case, $\varepsilon_3; \varepsilon_2$ must also be independent, and doing that swap instead, produces another arrow in the same equivalence class as $[\varepsilon_4; \varepsilon_2']$. We now get the functor

$$M : \mathcal{G} \to \mathcal{T}$$

Again, $\mathcal{T}$ has as objects graphs, and as arrows elementary event structures. In general, composable pairs of rewrites scattered through an arrow θ of $\mathcal{G}$, will correspond to causally related pairs with no intervening event caused by the first and causing the second. Such pairs will also be independent of the same sets of other events, and lead to a well defined notion of smallest elementary event structure corresponding to θ, obtained when all possible such internal compositions are performed after appropriate permutations of rewrites in θ. This smallest elementary event structure is the arrow of $\mathcal{T}$ corresponding to θ. We eventually find,

$$\begin{array}{ccccc}
\mathcal{T}^\infty & \xrightarrow{B_M} & \mathcal{T} & \xrightarrow{C_M} & \mathcal{T}_C \\
\uparrow{\scriptstyle M^\infty} & & \uparrow{\scriptstyle M} & & \uparrow{\scriptstyle M_C} \\
\mathcal{G}^\infty & \xrightarrow[B]{} & \mathcal{G} & \xrightarrow[C]{} & \mathcal{C}
\end{array}$$

where in the RHS square, the non-identity arrows of $\mathcal{T}_C$ are one-event elementary event structures corresponding to the single rewrites of arrows of $\mathcal{C}$, and C_M and M_C are the obvious functors.

So far our event semantics has had no need for notions of conflict, which is appropriate for considering a single arrow and its equivalence class. Now starting first with $\mathcal{G}^\infty$ as before, let $\Theta : G \to H_1$ and $\Psi : G \to H_2$ be equivalence classes of arrows of $\mathcal{G}^\infty$, with corresponding arrows $\mathrm{E} = (E, \leq_E) : G \to H_1$ and $\mathrm{F} = (F, \leq_F) : G \to H_2$ of $\mathcal{T}^\infty$. We define the relation on events in E and F by

$$e \approx f \text{ iff there is a } \theta \in \Theta \text{ and } \psi \in \Psi \text{ such that } \theta \text{ and } \psi$$
$$\text{have a common prefix } \phi, \text{ and } e(\theta) = f(\psi) \leq \mathrm{length}(\phi)$$

(Here "common prefix" refers to causal order rather than textual order in arrows of $\mathcal{G}^\infty$.) Clearly $\approx$ extends to an isomorphism of left-closed substructures of E and F. What we really want is the "colimit" of E and F along this common part. Now the events in E and F are all distinct due to the quantity of information packed within them, so $E \cap F = \emptyset$ unless $\Theta = \Psi$. We exploit this in order to make a construction without the usual tags. Let

$X = (E \cup F)/\approx$
$x \leq_X y$ iff $\exists e \in x, f \in y$, such that $e \leq_E f$ or $e \leq_F f$
$x \#_X y$ iff $x \subseteq E$ and $y \subseteq F$, or $x \subseteq F$ and $y \subseteq E$

$\mathrm{X} = (X, \leq_X, \#_X)$ is now a prime event structure, and something similar works for larger sets of equivalence classes of coinitial arrows of $\mathcal{G}^\infty$. For $\mathcal{G}$, a similar construction works, and is most easily obtained by restricting consideration to those executions that are interleavings of precisely the minimal elementary event structures that are the arrows of $\mathcal{T}$.

5 Graph Grammars and Good Granularity

A graph grammar $\mathcal{R}$ is a pair (R, G_0) where R is a (finite) collection of DPO rules, and G_0 is an initial graph. We can consider our preceding constructions in the light of a specific grammar $\mathcal{R}$. By $\mathcal{P}_\mathsf{R}$ we denote the subcategory of $\mathcal{P}$ consisting of (canonical rules corresponding to) rules of R, and whatever compositions and additional identities are required. Similarly we get $\mathcal{G}(\mathcal{P}, \mathcal{R}ew)_\mathsf{R}$ consisting of those rewrites generated by $\mathcal{P}_\mathsf{R}$ on arbitrary canonical redexes. We can now construct $\mathcal{G}_\mathsf{R}^{\infty *}$ consisting of those rewriting sequences where all rules used are from $\mathcal{P}_\mathsf{R}$. This will contain a subcategory $\mathcal{G}_\mathcal{R}^{\infty *}$ consisting of graphs derivable from (the skeletal graph of) G_0 using rules from $\mathcal{P}_\mathsf{R}$, and $\mathcal{G}_\mathcal{R}^{\infty *}$ will contain a subcategory $\mathcal{G}_\mathcal{R}^{\infty}$ where all rules used in arrows are just those from R, excluding any extraneous compositions or identities. There will be a subcategory $\mathcal{G}(\mathcal{P}, \mathcal{R}ew)_\mathcal{R}$ of $\mathcal{G}(\mathcal{P}, \mathcal{R}ew)_\mathsf{R}$ consisting of just those rewrites that contribute to $\mathcal{G}_\mathcal{R}^{\infty *}$. The categories $\mathcal{G}_\mathsf{R}$ and $\mathcal{G}_\mathcal{R}$ are got from $\mathcal{G}_\mathsf{R}^{\infty *}$ and $\mathcal{G}_\mathcal{R}^{\infty *}$ by doing all internal compositions; while the categories $\mathcal{C}_\mathsf{R}$ and $\mathcal{C}_\mathcal{R}$ contain the single rewrite equivalent of all derivations using only rules in $\mathcal{P}_\mathsf{R}$ or R (the distinction now disappears), and all derivations of $\mathcal{R}$ respectively. The various functors considered in section 3 restrict in a predictable way to the subcategories we have mentioned.

The event semantics of section 4 now supplies us with natural abstract semantics for a graph grammar $\mathcal{R}$. We consider all the arrows of $\mathcal{G}_\mathcal{R}^{\infty *}$, $\mathcal{G}_\mathcal{R}^{\infty}$ and $\mathcal{G}_\mathcal{R}$ with domain G_0, and thence construct the associated event structures $\mathrm{E}_\mathcal{R}^{\infty *} = (E_\mathcal{R}^{\infty *}, \leq_\mathcal{R}^{\infty *}, \#_\mathcal{R}^{\infty *})$, $\mathrm{E}_\mathcal{R}^{\infty} = (E_\mathcal{R}^{\infty}, \leq_\mathcal{R}^{\infty}, \#_\mathcal{R}^{\infty})$, and $\mathrm{E}_\mathcal{R} = (E_\mathcal{R}, \leq_\mathcal{R}, \#_\mathcal{R})$, the configuration spaces of which will be finitary prime algebraic domains.

Definition 5.1 A graph grammar $\mathcal{R} = (\mathsf{R}, G_0)$ has good granularity, iff no rule of R is composable with itself or with any other rule of R.

It is clear that the overwhelming majority of graph grammars used in practice will have good granularity. The main consequence of good granularity for a graph grammar $\mathcal{R}$, is that the restriction of the functor B which it induces, namely $B_\mathcal{R} : \mathcal{G}_\mathcal{R}^{\infty} \to \mathcal{G}_\mathcal{R}$, is an isomorphism between $\mathcal{G}_\mathcal{R}^{\infty}$ and $\mathcal{G}_\mathcal{R}$, because there are no internal compositions. This in turn forces an isomorphism between $\mathrm{E}_\mathcal{R}^{\infty}$ and $\mathrm{E}_\mathcal{R}$.

6 Discussion

In the preceding sections, we have reappraised the classical DPO construction within the opfibration framework. Subsequently we constructed various rewriting categories on the basis of a number of ways of combining rules and/or rewrites. Then we reinterpreted (equivalence classes under independence of) the arrows of these categories as elementatry event structures, leading to the further interpretation of suitable collections of coinitial arrows as event structures with conflict. At the end we focused on specific graph grammars to derive an abstract semantics for graph grammars via this route. Some comments are now in order.

To construct an opfibration, we need both determinism and associativity. These properties are usually in conflict: determinism comes most naturally with a concrete nonassociative rewriting construction, while associativity comes most naturally when one considers rewriting up to isomorphism. We overcame this by using canonical redexes constructed from a specific skeleton category. This is reasonable, even when graphs have nontrivial automorphisms, since then, up to isomorphism, the choice of one result for a rewrite is no better or worse than that of another which differs from the first by an automorphism. In this way we obtained the best of both worlds, in the process making some later constructions simpler. Nevertheless such a choice of strategy has specific consequences later on, particularly as regards the treatment of autoconcurrency and similar phenomena. For instance, given an autoconcurrent rule, and two successive rewrites using it, it is easy enough to distinguish the effects of the two individual rewrites in a concrete nonassociative rewriting construction, but when rewriting is via canonical redexes, there is no way of telling them apart, and one is forced to regard them as causally dependent if the individuality of events is not to be undermined, despite the fact that this might be felt to be counterintuitive. In order to analyse this carefully, we had to be quite precise about what an event was in our general framework, and under what circumstances two events could be regarded as distinct.

Our work in sections 3 and 4 bears comparison with other papers, notably [4–9,13]. Unlike [4–8], we use a skeleton category rather than equivalence classes of graphs/derivations. Of course, due to the equivalence between skeleton categories and ones constructed using suitable equivalences (see eg. [4,5]), all that we have done has its counterpart in the alternative approach. The comparison with Schied's work is also instructive. Schied's concrete rewriting construction has aspects of both our concrete and abstract constructions. It is deterministic and carries some history in the structure of nodes while also being associative. The price one pays for this is the need to be confined to the graphs generated from a given start graph by a given grammar in order for the construction to satisfy Schied's definiteness condition. We overcame the same hurdle in a general setting by explicit choice of canonical entities. More generally, being based upon a "global" construction, our later constructions were not limited by the need for restriction to any particular class of graphs or of grammars, eg. the safe or the consumptive grammars.

References

[1] Banach R., *A Fibration Semantics for Extended Term Graph Rewriting.* in: Term Graph Rewriting: Theory and Practice. Sleep, Plasmeijer, van Eekelen (eds.), John Wiley, (1993), 91-100.

[2] Banach R., *Term Graph Rewriting and Garbage Collection Using Opfibrations.* Theor. Comp. Sci. **131**, (1994), 29-94.

[3] Barr M., Wells C., *Category Theory for Computing Science.* Prentice-Hall, (1990).

[4] Corradini A., Ehrig H., Löwe M., Montanari U., Rossi F., *Note on Standard Representations of Graphs and Graph Derivations.* in: Graph Transformations in Computer Science. Schneider, Ehrig (eds.), L.N.C.S. **776**, (1993), 104-118.

[5] Corradini A., Ehrig H., Löwe M., Montanari U., Rossi F., *Abstract Graph Derivations in the Double pushout Approach.* in: Graph Transformations in Computer Science. Schneider, Ehrig (eds.), L.N.C.S. **776**, (1993), 86-103.

[6] Corradini A., Ehrig H., Löwe M., Montanari U., Rossi F., *An Event Structure Semantics for Safe Graph Grammars.* in: Proc. I.F.I.P. Working Conference PROCOMET-94, to appear.

[7] Corradini A., Ehrig H., Löwe M., Montanari U., Rossi F., *An Event Structure Semantics for Consumptive Graph Grammars with Parallel Productions.* in: Proc. Fifth International Workshop on Graph Grammars and their Application to Computer Science 1994, Williamsburg U.S.A., L.N.C.S., to appear.

[8] Corradini A., Ehrig H., Löwe M., Montanari U., Padberg J., *Functorial Semantics for Safe Graph Grammars using Prime Algebraic Domains and Event Structures.* in: Proc. Fifth International Workshop on Graph Grammars and their Application to Computer Science 1994, Williamsburg U.S.A., L.N.C.S. to appear.

[9] Corradini A., Ehrig H., Löwe M., Montanari U., Padberg J., *Typed Graph Grammars and their Adjunction with Categories of Derivations.* in: Proc. Fifth International Workshop on Graph Grammars and their Application to Computer Science 1994, Williamsburg U.S.A., L.N.C.S. to appear.

[10] Ehrig H., *Introduction to the Algebraic Theory of Graph Grammars (A survey).* in: L.N.C.S. **73**, (1979), 1-69.

[11] Ehrig H., *A Tutorial Introduction to the Algebraic Approach of Graph Grammars.* in: Third International Workshop on Graph Grammars, L.N.C.S. **291**, (1986), 3-14.

[12] Nielsen M., Plotkin G., Winskel G., *Petri Nets, Event Structures and Domains, Part I.* Theor. Comp. Sci. **13**, (1981), 85-108.

[13] Schied G., *On Relating Rewriting Systems and Graph Grammars to Event Structures.* in: Graph Transformations in Computer Science, Schneider, Ehrig (eds.), L.N.C.S. **776**, (1993), 326-340.