Domains and Event Structures for Fusions

Paolo Baldan
University of Padova
Andrea Corradini, Fabio Gadducci
University of Pisa

***Abstract*—Stable event structures, and their duality with prime algebraic domains (arising as partial orders of configurations), are a landmark of concurrency theory, providing a clear characterisation of causality in computations. They have been used for defining a concurrent semantics of several formalisms, from Petri nets to linear graph rewriting systems, which in turn lay at the basis of many visual frameworks. Stability however is restrictive for dealing with formalisms where a computational step can merge parts of the state, like graph rewriting systems with non-linear rules, which are needed to cover some relevant applications (such as the graphical encoding of calculi with name passing). We characterise, as a natural generalisation of prime algebraic domains, a class of domains that is well-suited to model the semantics of formalisms with fusions. We then identify a corresponding class of event structures, that we call connected event structures, via a duality result formalised as an equivalence of categories. We show that connected event structures are exactly the class of event structures that arise as the semantics of non-linear graph rewriting systems. Interestingly, the category of general unstable event structures coreflects into our category of domains, so that our result provides a characterisation of the partial orders of configurations of such event structures.**

***Index Terms*—Event structures, fusions, graph rewriting, process calculi.**

I. Introduction

For a long time stable/prime event structures and their duality with prime algebraic domains have been considered one of the landmarks of concurrency theory, providing a clear characterisation of causality in software systems. They have been used to provide a concurrent semantics to a wide range of foundational formalisms, from Petri nets [1] to linear graph rewriting systems [2]–[4] and process calculi [5]–[7]. They are one of the standard tools for the formal treatment of (true, i.e., non-interleaving) concurrency. See, e.g., [8] for a reasoned survey on the use of such causal models. Recently, they have been used in the study of concurrency in weak memory models [9], [10] and for process mining and differencing [11].

In order to endow a chosen formalism with an event structure semantics, a standard construction consists in viewing the class of computations as a partial order. An element of the order is some sort of configuration, i.e., an execution trace up to an equivalence that identifies traces differing only for the order of independent steps (e.g., interchange law [12] in term rewriting, shift equivalence [13] in graph rewriting, permutation equivalence [14] in the lambda-calculus, ...), and the order relates two computations when the latter is an extension of the former. Events are then identified with

978-1-5090-3018-7/17/$31.00

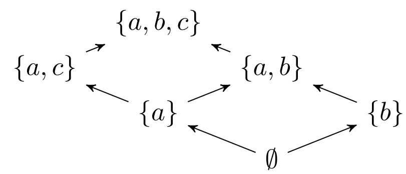
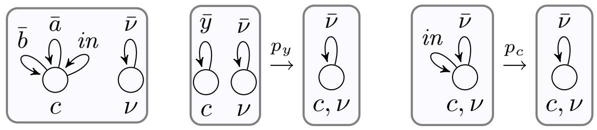


Fig. 1: The domain of configurations of the process $a.c \mid b$.

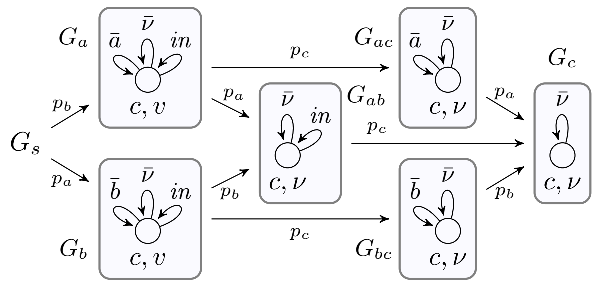
configurations consisting of a maximal computation step (e.g., a transition of a CCS process or a transition firing for a Petri net) with all its causes. As a simple example, consider the CCS process $a.c \mid b$. The corresponding partial order is depicted in Fig. 1. The events correspond to configurations $\{a\}$ (transition a with empty set of causes), $\{a, c\}$ (transition c caused by a), and $\{b\}$ (transition b with empty set of causes). The fact that each event in a configuration has a uniquely determined set of causes, a property that for event structures is called *stability*, allows one to characterise such elements, order theoretically, as the prime elements: if they are included in a join they must be included in one of the joined elements. Each element of the partial order of configurations can be reconstructed uniquely as the join of the primes so that the partial order is prime algebraic. This duality between event structures and domains of configurations can be nicely formalised in terms of an equivalence between the category of prime event structures and that of prime algebraic domains [1], [15].

The set up described so far fails when moving to formalisms where a computational step can merge parts of the state, as it happens whenever we consider nominal calculi where as a result of name passing the received name is identified with a local one at the receiver [16], [17] or in the modelling of bonding in biological/chemical processes [18]. Whenever we think of the state of the system as some kind of graph with the dynamics described by graph rewriting, this means that rules are non-linear (more precisely, in the so-called double pushout approach [19], left-linear but possibly not right-linear). In general terms, the point is that in the presence of fusions the same event can be enabled by different minimal sets of events, thus preventing the identification of a notion of causality.

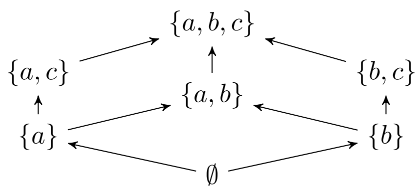
As an example, consider the graph rewriting system in Fig. 2. Figure 2a reports the start graph G_s and the rewriting rules p_a, p_b, and p_c. Observe that rules p_y, where y can be either a or b, delete edge $\bar{y}$ and merge nodes c and ν. The possible rewrites are depicted in Fig. 2b. For instance, applying p_a



(a) The start graph G_s and the rules p_y ($y \in \{a, b\}$) and p_c



(b) The possible rewrites



(c) The domain of configurations

Fig. 2: A graph rewriting system with fusions.

to G_s we get the graph G_b. Now, p_b can still be applied to G_b matching its left-hand side non-injectively, thus getting graph G_{ab}. Similarly, we can apply first p_b and then p_a, obtaining again G_{ab}. Observe that at least one of p_a and p_b must be applied to enable p_c, since the latter rule requires nodes c and ν to be merged. The graph rewriting system is a (simplified) representation of the π-calculus process $(\nu c)(\bar{a}(c) \mid \bar{b}(c) \mid c())$. Rules p_y, for $y \in \{a, b\}$, represent the execution of $\bar{y}(c)$ that outputs on channel y the restricted name c. The first rule that is executed extrudes name c, while the second is just a standard output. Only after the extrusion the name c is available outside the scope and the input prefix $c()$ can be consumed. Observe that in a situation where all the three rules p_a, p_b, and p_c are applied, since p_a and p_b are independent, it is not possible to define a proper notion of causality. We only know that at least one of p_a and p_b must be applied before p_c. The corresponding domain of configurations, reported in Fig. 2c, is naturally derived from the possible rewrites in Fig. 2b.

The impossibility of modelling these situations with stable event structures is well-known (see, e.g., [15] for a general discussion, [2] for graph rewriting systems or [16] for the π-calculus). One has to drop the stability requirement and replace causality by an enabling relation $\vdash$. More precisely, in the specific case we would have $\emptyset \vdash a$, $\emptyset \vdash b$, $\{a\} \vdash c$, $\{b\} \vdash c$.

The questions that we try to answer is: what can be retained of the satisfactory duality between events structures and domains, when dealing with formalisms with fusions? Which are the properties of the domain of computations that arise in this setting? What are the event structure counterparts?

The domain of configurations of the example suggests that in this context an event is still a computation that cannot be decomposed as the join of other computations hence, in order theoretical terms, it is an irreducible. However, due to unstability, irreducibles are not primes: two different irreducibles can be different minimal histories of the same event, in a way that an irreducible can be included in a computation that is the join of two computations without being included in any of the two. For instance, in the example above, $\{a, c\}$ is an irreducible, corresponding to the execution of c enabled by a, and it is included in $\{a\} \sqcup \{b, c\} = \{a, b, c\}$, although neither $\{a, c\} \subseteq \{a\}$ nor $\{a, c\} \subseteq \{b, c\}$. Uniqueness of decomposition of an element in terms of irreducibles also fails, e.g., $\{a, b, c\} = \{a\} \sqcup \{b\} \sqcup \{a, c\} = \{a\} \sqcup \{b\} \sqcup \{b, c\}$: the irreducibles $\{a, c\}$ and $\{b, c\}$ can be used interchangeably in the decomposition of $\{a, b, c\}$.

Building on the previous observation, we introduce an equivalence on irreducibles identifying those that can be used interchangeably in the decompositions of an element (intuitively, different minimal histories of the same event). Based on this we give a weaker notion of primality (i.e., up to interchangeability) such that the class of domains suited for modelling the semantics of formalisms with fusions are defined as the class of weak prime algebraic domains.

Given a weak prime algebraic domain, a corresponding event structure can be obtained by taking as events the set of irreducibles, quotiented under the (transitive closure of the) interchangeability relation. The resulting class of event structures is a (mild) restriction of the general unstable event structures in [15] that we call connected event structures. Categorically, we get an equivalence between the category of weak prime algebraic domains and the one of connected event structures, generalising the equivalence between prime algebraic domains and prime event structures.

We also show that, in the same way as prime algebraic domains/prime event structures are exactly what is needed for Petri nets/linear graph rewriting systems, weak prime algebraic domains/connected event structures are exactly what is needed for non-linear graph rewriting systems: each rewriting system maps to a connected event structure and conversely each connected event structure arises as the semantics of some rewriting system. This supports the adequateness of weak prime algebraic domains and connected event structures as semantics structures for formalisms with fusions.

Interestingly, we can also show that the category of general unstable event structures [15] coreflects into our category of weak prime algebraic domains. Therefore our notion of weak prime algebraic domain can be seen as a novel characterisation of the partial order of configurations of such event structures that is alternative to those based on intervals in [20], [21]. Our characterisation is a natural generalisation of the one for prime event structures, with irreducibles (instead of primes) having a tight connection with events. The correspondence is established at a categorical level, as a coreflection of categories, something that, to the best of our knowledge, had not been done before in the literature.

The rest of the paper is structured as follows. In Section II we recall the basics of (prime) event structures and their correspondence with prime algebraic domains. In Section III we introduce weak prime algebraic domains, connected event structures and establish a duality result. In Section IV we show the intimate connection between weak prime algebraic domains, or equivalently connected event structures, and non-linear graph rewriting systems. Finally, in Section V we wrap up the main contributions of the paper and we sketch further advances and possible connections with related works.

II. Background: domains and event structures

This section recalls the notion of event structures, as introduced in [15], and their duality with partial orders.

A. Event structures

In the paper, we focus on event structures with binary conflict. This choice plays a role in the relation with graph rewriting (Section IV), while the duality results in Section III could be easily rephrased for non-binary conflicts expressed by means of a consistency predicate. Given a set X we denote by $\mathbf{2}^X$ and $\mathbf{2}^X_{\mathit{fin}}$ the powerset and the set of finite subsets of X, respectively. For $m, n \in \mathbb{N}$, we denote by $[m, n]$ the set $\{m, m+1, \ldots, n\}$.

Definition 1 (event structure) *An* event structure *(*ES *for short) is a tuple $\langle E, \vdash, \# \rangle$ such that*

- *E is a set of events;*
- *$\vdash \subseteq \mathbf{2}^E_{\mathit{fin}} \times E$ is the* enabling *relation satisfying $X \vdash e$ and $X \subseteq Y$ implies $Y \vdash e$;*
- *$\# \subseteq E \times E$ is the conflict relation.*

An $X \subseteq E$ is consistent *if there is no $e, e' \in X$ with $e \# e'$.*

An ES $\langle E, \vdash, \# \rangle$ is often denoted simply by E. Computations are captured by the notion of configuration.

Definition 2 (configuration, live ES) *A* configuration *of an* ES *E is a consistent $C \subseteq E$ which is* secured*, i.e., for all $e \in C$ there are $e_1, \ldots, e_n \in C$ with $e_n = e$ such that $\{e_1, \ldots, e_{k-1}\} \vdash e_k$ for all $k \in [1, n]$ (in particular, $\emptyset \vdash e_1$). The set of configurations of an* ES *E is denoted by $\mathit{Conf}(E)$ and the subset of* finite *configurations by $\mathit{Conf}_F(E)$. An* ES *is* live *if conflict is saturated, i.e., for all $e, e' \in E$ if there is no $C \in \mathit{Conf}(E)$ such that $\{e, e'\} \subseteq C$ then $e \# e'$ and moreover for all $e \in E$ we have $\neg(e \# e)$.*

In this setting, two events are *concurrent* when they are consistent and enabled by the same configuration.

Remark 3 *In the rest of the paper we restrict to live* ES*, where conflict is saturated (this corresponds to inheritance of conflict in prime event structures) and each event is executable. Hence the qualification live is omitted.*

Since the enabling predicate is over finite sets of events, we can consider minimal sets of events enabling a given one.

Definition 4 (minimal enabling) *Given an* ES *$\langle E, \vdash, \# \rangle$ define $C \vdash_0 e$ when $C \in \mathit{Conf}(E)$, $C \vdash e$ and for any other configuration $C' \subseteq C$, if $C' \vdash e$ then $C' = C$.*

The classes of stable and prime ES represent our starting point and play an important role in the paper.

Definition 5 (stable and prime ES) *An* ES *$\langle E, \vdash, \# \rangle$ is* stable *if $X \vdash e$, $Y \vdash e$, and $X \cup Y \cup \{e\}$ consistent imply $X \cap Y \vdash e$. It is* prime *if $X \vdash e$ and $Y \vdash e$ imply $X \cap Y \vdash e$.*

For stable ES, given a configuration C and an event $e \in C$, there is a unique minimal configuration $C' \subseteq C$ such that $C' \vdash_0 e$. The set C' can be seen as the set of causes of the event e in the configuration C. This gives a well-defined notion of causality that is local to each configuration. In a prime ES, for any event e there is a unique minimal enabling $C \vdash_0 e$, thus providing a global notion of causality. In general, in possibly unstable ES, due to the presence of consistent *or-enablings*, there might be distinct minimal enablings in the same configuration.

Example 6 *A simple example of unstable* ES *is the following: the set of events is $\{a, b, c\}$, the conflict relation $\#$ is the empty one and the minimal enablings are $\emptyset \vdash_0 a$, $\emptyset \vdash_0 b$, $\{a\} \vdash_0 c$, and $\{b\} \vdash_0 c$. Thus, event c has two minimal enablings and these are consistent, hence $\{a, b\} \vdash c$. This is the event structure discussed in the introduction, whose domain of configurations is reported in Fig. 2c.*

The class of ES can be turned into a category.

Definition 7 (category of ES) *A morphism of* ES *$f : E_1 \to E_2$ is a partial function $f : E_1 \to E_2$ such that for all $X_1 \subseteq E_1$ and $e_1, e_1' \in E_1$ with $f(e_1)$, $f(e_1')$ defined*

- *if $f(e_1) \# f(e_1')$ then $e_1 \# e_1'$*
- *if $f(e_1) = f(e_1')$ and $e_1 \neq e_1'$ then $e_1 \# e_1'$;*
- *if $X_1 \vdash_1 e_1$ then $f(X_1) \vdash_2 f(e_1)$.*

We denote by ES *the category of* ES *and their morphisms and by* sES *and* pES *the full subcategories of stable and prime* ES.

B. Domains

A preordered or partially ordered set $\langle D, \sqsubseteq \rangle$ is often denoted simply as D, omitting the (pre)order relation. We denote by $\preceq$ the immediate predecessor relation, i.e., $x \preceq y$ whenever $x \sqsubseteq y$ and for all z if $x \sqsubseteq z \sqsubseteq y$ then $z \in \{x, y\}$. A subset $X \subseteq D$ is *consistent* if it has an upper bound $d \in D$ (i.e., $x \sqsubseteq d$ for all $x \in X$). It is *pairwise consistent* if every two elements subset of X is consistent. A subset $X \subseteq D$ is *directed* if $X \neq \emptyset$ and every pair of elements in X has an upper bound in X. It is an *ideal* if it is directed and downward closed. Given an element $x \in D$, we write $\downarrow x$ to denote the *principal ideal* $\{y \in D \mid y \sqsubseteq x\}$ generated by x. Given a partial order D, its ideal completion, denoted by $\mathsf{Idl}(D)$, is the set of ideals of D, whose order is given by subset inclusion. The *least upper bound* and the *greatest lower bound* of a subset $X \subseteq D$ (if they exist) are denoted by $\bigsqcup X$ and $\bigsqcap X$, respectively.

Definition 8 (domains) *A partial order D is* coherent *if for all pairwise consistent $X \subseteq D$ the least upper bound $\bigsqcup X$ exists. An element $d \in D$ is* compact *if for all directed $X \subseteq D$ $d \sqsubseteq \bigsqcup X$ implies $d \sqsubseteq x$ for some $x \in X$. The set of compact elements of D is denoted by $\mathsf{K}(D)$. A coherent partial order D is* algebraic *if for every $x \in D$ we have $x = \bigsqcup(\downarrow x \cap \mathsf{K}(D))$. We say that D is* finitary *if for every element $a \in \mathsf{K}(D)$ the set $\downarrow a$ is finite. We refer to algebraic finitary coherent partially ordered sets as* domains.

Note that in a domain all non-empty subsets have a meet. In fact, if $\emptyset \neq X \subseteq D$, then $\bigsqcap X = \bigsqcup L(X)$ where $L(X) = \{y \mid \forall x \in X.\, y \sqsubseteq x\}$ is the set of lowerbounds of X that is pairwise consistent since it is dominated by any $x \in X$.

For a domain D we can think of its elements as "pieces of information" expressing the states of evolution of a process. Compact elements represent states that are reached after a finite number of steps. Thus algebraicity essentially says that any infinite computation can be approximated with arbitrary precision by the finite ones. More formally, when D is algebraic it is determined by $\mathsf{K}(D)$, i.e., $D \simeq \mathsf{Idl}(\mathsf{K}(D))$.

For an ES, the configurations ordered by subset inclusion form a domain. When the ES is stable, if an event with its minimal history is in the join of different configurations, then it belongs, with the same history, to one of such configurations. In order-theoretic terms, minimal histories are prime elements, representing the building blocks of computations.

Definition 9 (primes and prime algebraicity) *Let D be a domain. A* prime *is an element $p \in \mathsf{K}(D)$ such that, for any pairwise consistent $X \subseteq \mathsf{K}(D)$, if $p \sqsubseteq \bigsqcup X$ then $p \sqsubseteq x$ for some $x \in X$. The set of prime elements of D is denoted by $pr(D)$. The domain D is* prime algebraic *(or simply* prime*) if for all $x \in D$ we have $x = \bigsqcup(\downarrow x \cap pr(D))$.*

Prime domains are the domain theoretical counterpart of stable and prime ES. For a stable ES $\langle E, \#, \vdash\rangle$, the partial order $\langle \mathit{Conf}(E), \subseteq\rangle$ is a prime domain, denoted $\mathcal{D}_S(E)$. Conversely, given a prime domain D, the triple $\langle pr(D), \#, \vdash\rangle$, where $p\#p'$ if $\{p, p'\}$ is not consistent and $X \vdash p$ when $(\downarrow p \cap pr(D)) \setminus \{p\} \subseteq X$, is a prime ES, denoted $\mathcal{E}_S(D)$.

This correspondence can be elegantly formulated at the categorical level [15]. We recall the notion of domain morphism.

Definition 10 (category of prime domains) *Let D_1, D_2 be prime domains. A morphism $f : D_1 \to D_2$ is a total function such that for all $X_1 \subseteq D_1$ consistent and $d_1, d_1' \in D_1$*

1) if $d_1 \preceq d_1'$ then $f(d_1) \preceq f(d_1')$;
2) $f(\bigsqcup X_1) = \bigsqcup f(X_1)$;
3) if $X_1 \neq \emptyset$ then $f(\bigsqcap X_1) = \bigsqcap f(X_1)$;

We denote by pDom *the category of prime domains and their morphisms.*

The correspondence is then captured by the result below.

Theorem 11 (duality) *There are functors $\mathcal{D}_S : \mathsf{sES} \to \mathsf{pDom}$ and $\mathcal{E}_S : \mathsf{pDom} \to \mathsf{sES}$ establishing a coreflection. It restricts to an equivalence of categories between* pDom *and* pES.

III. Weak prime domains and connected ES

In this section we characterise a class of domains, and the corresponding brand of ES, that are suited for expressing the semantics of computational formalisms with fusions.

A. Weak prime algebraic domains

We show that domains arising in the presence of fusions are characterised by resorting to a weakened notion of prime element. We start recalling the notion of irreducible element.

Definition 12 (irreducibles) *Let D be a domain. An* irreducible *of D is an element $i \in \mathsf{K}(D)$ such that, for any pairwise consistent $X \subseteq \mathsf{K}(D)$, if $i = \bigsqcup X$ then $i \in X$. The set of irreducibles of D is denoted by $ir(D)$ and, for $d \in D$, we define $ir(d) = \downarrow d \cap ir(D)$.*

Irreducibles in domains have a simple characterisation.

Lemma 13 (unique predecessor for irreducibles) *Let D be a domain and $i \in D$. Then $i \in ir(D)$ iff it has a unique immediate predecessor, denoted $p(i)$.*

We next observe that any domain is actually irreducible algebraic, namely it can be generated by the irreducibles.

Proposition 14 (domains are irreducible algebraic) *Let D be a domain. Then for any $d \in D$ it holds $d = \bigsqcup ir(d)$.*

Now note that any prime is an irreducible. If D is a prime domain then also the converse holds, i.e., the irreducibles coincide with the primes.

Proposition 15 (irreducibles vs. primes) *Let D be a domain. Then D is a prime domain iff $pr(D) = ir(D)$.*

Quite intuitively, in the domain of configurations of an ES the irreducibles are minimal histories of events. For instance, in the domain depicted in Fig. 2c the irreducibles are $\{a\}$, $\{b\}$, $\{a, c\}$, and $\{b, c\}$. For stable ES, the domain is prime and thus, as observed above, irreducibles coincide with primes. This fails in unstable ES, as we can see in our running example: while $\{a\}$ and $\{b\}$ are primes, the two minimal histories of c, namely $\{a, c\}$ and $\{b, c\}$, are not. In fact, $\{a, c\} \subseteq \{a\} \sqcup \{b, c\}$, but neither $\{a, c\} \subseteq \{a\}$ nor $\{a, c\} \subseteq \{b, c\}$.

The key observation is that in general an event corresponds to a class of irreducibles, like $\{a, c\}$ and $\{b, c\}$ in our example. Additionally, two irreducibles corresponding to the same event can be used, to a certain extent, interchangeably for building the same configuration. For instance, $\{a, b, c\} = \{a, b\} \sqcup \{a, c\} = \{a, b\} \sqcup \{b, c\}$. We next formalise this intuition, i.e., we interpret irreducibles in a domain as minimal histories of some event and we identify classes of irreducibles corresponding to the same event.

We start by observing that in a prime domain any element admits a unique decomposition in terms of irreducibles.

Lemma 16 (unique decomposition) *Let D be a prime domain. If $X, X' \subseteq ir(D)$ are downward closed sets of irreducibles such that $\bigsqcup X = \bigsqcup X'$ then $X = X'$.*

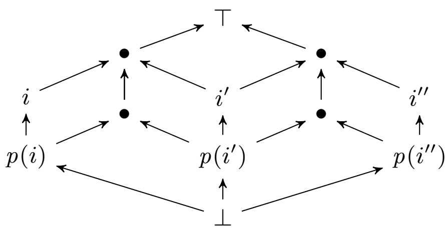


Fig. 3: Interchangeability need not be transitive.

The result above no longer holds in domains arising in the presence of fusions. For instance, in the domain in Fig. 2c, $X = \{\{a\},\{b\},\{a,c\}\}$, $X' = \{\{a\},\{b\},\{b,c\}\}$ and $X'' = \{\{a\},\{b\},\{b,c\},\{a,c\}\}$ are all decompositions for $\{a,b,c\}$. The idea is to identify irreducibles that can be used interchangeably in a decomposition.

Definition 17 (interchangeability) *Let D be a domain and $i, i' \in ir(D)$. We write $i \leftrightarrow i'$ if for all $X \subseteq ir(D)$ such that $X \cup \{i\}$ and $X \cup \{i'\}$ are downward closed and consistent we have $\bigsqcup(X \cup \{i\}) = \bigsqcup(X \cup \{i'\})$ and for some such X it holds $\bigsqcup X \neq \bigsqcup(X \cup \{i\})$.*

In words, $i \leftrightarrow i'$ means that i and i' produce the same effect when added to a decomposition that already includes their predecessors and there is at least one situation in which the addition of i and i' produces some effect. Hence, intuitively, i and i' correspond to the execution of the same event.

We now give some characterisations of interchangeability.

Lemma 18 (characterising $\leftrightarrow$) *Let D be a domain and $i, i' \in ir(D)$. Then the following are equivalent*

1) *$i \leftrightarrow i'$;*
2) *i, i' are consistent and for all $d \in \mathsf{K}(D)$ such that $p(i), p(i') \sqsubseteq d$, $d \sqcup i = d \sqcup i'$ and for some such d it holds $d \neq d \sqcup i$;*
3) *i, i' are consistent and $i \sqcup p(i') = p(i) \sqcup i' \neq p(i) \sqcup p(i')$.*

The interchangeability relation is reflexive and symmetric, but not transitive: in the domain of Fig. 3, $i \leftrightarrow i'$ and $i' \leftrightarrow i''$ but not $i \leftrightarrow i''$. The same holds in the domain obtained by removing the top element.

We now introduce weak primes: they weaken the property of prime elements, requiring that it holds up to interchangeability.

Definition 19 (weak prime) *Let D be a domain. A* weak prime *of D is an element $i \in ir(D)$ such that for any consistent $X \subseteq D$, if $i \sqsubseteq \bigsqcup X$ then there exist $i' \in ir(D)$ with $i \leftrightarrow i'$ and $d \in X$ such that $i' \sqsubseteq d$. We denote by $wpr(D)$ the set of weak primes of D.*

Clearly, since interchangeability is reflexive, any prime is a weak prime. Moreover, in prime domains also the converse holds as interchangeability turns out to be the identity.

Lemma 20 (weak primes in prime domains) *Let D be a prime domain. Then $\leftrightarrow$ is the identity and $wpr(D) = pr(D)$.*

We argue that the domain of configurations arising in the presence of fusions can be characterised domain-theoretically by asking that all irreducibles are weak primes, i.e., that the domain is algebraic with respect to weak primes.

Definition 21 (weak prime algebraic domains) *Let D be a domain. It is* weak prime algebraic *(or simply* weak prime*) if for any $d \in D$ it holds $d = \bigsqcup(\downarrow d \cap wpr(D))$.*

In the same way as prime domains are domains where all irreducibles are primes (see Proposition 15), we can provide a characterisation of weak prime domains in terms of coincidence between irreducibles and weak primes.

Proposition 22 (weak prime domains, again) *Let D be a domain. It is weak prime iff all irreducibles are weak primes.*

We finally introduce a category of weak prime domains by defining a notion of morphism.

Definition 23 (category of weak prime domains) *Let D_1, D_2 be weak prime domains. A weak prime domain morphism $f : D_1 \to D_2$ is a total function such that for all $X_1 \subseteq D_1$ consistent and $d_1, d_1' \in D_1$*

1) *if $d_1 \preceq d_1'$ then $f(d_1) \preceq f(d_1')$;*
2) *$f(\bigsqcup X_1) = \bigsqcup f(X_1)$;*
3) *if d_1, d_1' consistent and $d_1 \sqcap d_1' \preceq d_1$ then $f(d_1 \sqcap d_1') = f(d_1) \sqcap f(d_1')$;*

We denote by wDom *the category of weak prime domains and their morphisms.*

Compared with the notion of morphism for prime domains in Definition 10 (from [15]), we still require the preservation of $\preceq$ and $\sqcup$ of consistent sets (conditions (1) and (2)). However, the third condition, i.e., preservation of $\sqcap$, is weakened to preservation in some cases. General preservation of meets is indeed not expected in the presence of fusions. Consider e.g. the running example in Example 6 and another ES $E' = \{c\}$ with $\emptyset \vdash c$ and the morphism $f : E \to E'$ that forgets a and b. Then $f(\{a,c\}) \sqcap f(\{b,c\}) = \{c\} \sqcap \{c\} = \{c\} \neq f(\{a,c\} \sqcap \{b,c\}) = f(\emptyset) = \emptyset$. Intuitively, the condition $d_1 \sqcap d_1' \prec d_1$ means that d_1' includes the computation modelled by d_1 apart from a final step, hence $d_1 \sqcap d_1'$ coincides with d_1 when such step is removed. Since domain morphisms preserve immediate precedence (i.e., single steps), also $f(d_1)$ differs from $f(d_1')$ for the execution of a final step and the meet $f(d_1) \sqcap f(d_1')$ is $f(d_1)$ without such step, and thus it coincides with $f(d_1 \sqcap d_1')$.

In general we only have

$$f(\bigsqcap X_1) \sqsubseteq \bigsqcap f(X_1)$$

In fact, for all $x_1 \in X_1$, we have $\bigsqcap X_1 \sqsubseteq x_1$, hence $f(\bigsqcap X_1) \sqsubseteq f(x_1)$ and thus $f(\bigsqcap X_1) \sqsubseteq \bigsqcap f(X_1)$. Still, when restricted to prime domains, our notion of morphism boils down to the original one, i.e., the full subcategory of wDom having prime domains as objects is pDom.

Proposition 24 *The category of prime domains* pDom *is the full subcategory of* wDom *having prime domains as objects.*

B. Connected event structures

We show that the set of configurations of an ES, ordered by subset inclusion, is a weak prime domain where the compact elements are the finite configurations. Moreover, the correspondence can be lifted to a functor. We also identify a subclass of ES that we call *connected* ES and that are the exact counterpart of weak prime domains (in the same way as prime ES correspond to prime algebraic domains).

Definition 25 (partial order of configurations of an ES) *Let E be an ES. We denote by $\mathcal{D}(E)$ the partial order $\langle Conf(E), \subseteq \rangle$. Given an ES morphism $f : E_1 \to E_2$, its image $\mathcal{D}(f) : \mathcal{D}(E_1) \to \mathcal{D}(E_2)$ is defined as $\mathcal{D}(f)(C_1) = \{f(e_1) : e_1 \in C_1\}$.*

We first need some technical facts, collected in the following lemma. Recall that in the setting of unstable ES we can have distinct consistent minimal enablings for an event. When $C \vdash_0 e$, $C' \vdash_0 e$, and $C \cup C' \cup \{e\}$ is consistent, we write $C \overset{e}{\frown} C'$. We denote by $\overset{e}{\frown}{}^*$ the transitive closure of the relation $\overset{e}{\frown}$.

Lemma 26 *Let $\langle E, \vdash, Con \rangle$ be an ES. Then*

1) *$\mathcal{D}(E)$ is a domain, $\mathsf{K}(\mathcal{D}(E)) = Conf_F(E)$, join is union and $C \preceq C'$ iff $C = C' \cup \{e\}$ for some $e \in E$;*
2) *$C \in Conf_F(E)$ is irreducible iff $C = C' \cup \{e\}$ and $C' \vdash_0 e$; in this case we denote C as $\langle C', e \rangle$;*
3) *for $C \in Conf(E)$, we have $ir(C) = \{\langle C', e' \rangle \mid e' \in C \ \wedge \ C' \subseteq C \ \wedge \ C' \vdash_0 e'\}$;*
4) *for $\langle C_1, e_1 \rangle, \langle C_2, e_2 \rangle \in ir(\mathcal{D}(E))$, we have $\langle C_1, e_1 \rangle \leftrightarrow \langle C_2, e_2 \rangle$ iff $e = e_1 = e_2$ and $C_1 \overset{e}{\frown} C_2$.*

Concerning point 1, observe that the meet in the domain of configurations is $C \sqcap C' = \bigcup \{C'' \in Conf(E) \mid C'' \subseteq C \wedge C'' \subseteq C'\}$, which is usually smaller than the intersection. For instance, in Fig. 2, $\{a, c\} \sqcap \{b, c\} = \emptyset \neq \{c\}$. Point 2 says that irreducibles are configurations of the form $C \cup \{e\}$ that admits a secured execution in which the event e appears as the last one and cannot be switched with any other. In other words, irreducibles are minimal histories of events. Point 3 characterises the irreducibles in a configuration. According to point 4, two irreducibles are interchangeable when they are different histories for the same event.

Proposition 27 *Let E be an ES. Then $\mathcal{D}(E)$ is a weak prime domain. Given a morphism $f : E_1 \to E_2$, its image $\mathcal{D}(f) : \mathcal{D}(E_1) \to \mathcal{D}(E_2)$ is a weak prime domain morphism.*

A special role is played by the subclass of *connected* ES.

Definition 28 (connected es) *An* ES *is* connected *if whenever $C \vdash_0 e$ and $C' \vdash_0 e$ then $C \overset{e}{\frown}{}^* C'$. We denote by cES the full subcategory of ES having connected ES as objects.*

In words, different minimal enablings for the same event must be pairwise connected by a chain of consistency. For instance, the ES in Example 6 is a connected ES. Only event c has two minimal histories $\{a\} \vdash_0 c$ and $\{b\} \vdash_0 c$ and obviously $\{a\} \overset{c}{\frown} \{b\}$. Clearly, prime ES are also connected ES. More precisely, we have the following.

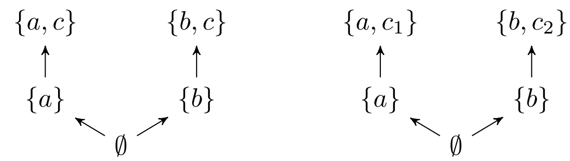


Fig. 4: Non-connected ES do not uniquely determine a domain.

Proposition 29 (connectedness, stability, primality) *Let E be an ES. Then E is prime iff it is stable and connected.*

The defining property of connected ES allows one to recognise that two minimal histories are relative to the same event by only looking at the partially ordered structure and thus, as we will see, from the domain of configurations of a connected ES we can recover an ES isomorphic to the original one and vice versa (see Theorem 35). In general, this is not possible. For instance, consider the ES E' with events $E' = \{a, b, c\}$, and where $a \# b$ and the minimal enablings are again $\emptyset \vdash_0 a$, $\emptyset \vdash_0 b$, $\{a\} \vdash_0 c$, and $\{b\} \vdash_0 c$. Namely, event c has two minimal enablings, but differently from what happens in the running example, these are not consistent, hence $\{a, b\} \not\vdash c$. The resulting domain of configurations is depicted on the left of Fig. 4. Intuitively, it is not possible to recognise that $\{a, c\}$ and $\{b, c\}$ are different histories of the same event. In fact, note that we would get an isomorphic domain of configurations by considering the ES E'' with events $E'' = \{a, b, c_1, c_2\}$ such that $a \# b$ and the minimal enablings are again $\emptyset \vdash_0 a$, $\emptyset \vdash_0 b$, $\{a\} \vdash_0 c_1$, and $\{b\} \vdash_0 c_2$.

C. From domains to event structures

We show how to get an ES from a weak prime domain. As expected, events are equivalence classes of irreducibles, where the equivalence is (the transitive closure of) interchangeability.

Domains are irreducible algebraic (see Proposition 14), hence any element is determined by the irreducibles under it. The difference between two elements is thus somehow captured by the irreducibles that are under one element and not under the other. This motivates the following definition.

Definition 30 (irreducible difference) *Let D be a domain and $d, d' \in \mathsf{K}(D)$ such that $d \sqsubseteq d'$. Then we define $\delta(d', d) = ir(d') \setminus ir(d)$.*

In a prime domain an element admits a unique decomposition in terms of primes (see Lemma 16). Here the same holds for irreducibles but only up to interchangeability. Given a domain D and an irreducible $i \in ir(D)$, we denote by $[i]_{\leftrightarrow^*}$ the corresponding equivalence class. For $X \subseteq ir(D)$ we define $[X]_{\leftrightarrow^*} = \{[i]_{\leftrightarrow^*} \mid i \in X\}$.

Proposition 31 (unique decomposition up to $\leftrightarrow$) *Let D be a weak prime domain, $d \in \mathsf{K}(D)$, and $X \subseteq ir(d)$ downward closed. Then $d = \bigsqcup X$ iff $[X]_{\leftrightarrow^*} = [ir(d)]_{\leftrightarrow^*}$.*

We now have the tools for mapping our domains to an ES.

Definition 32 (ES for a weak prime domain) *Let D be a weak prime domain. The* ES $\mathcal{E}(D) = \langle E, \#, \vdash \rangle$ *is defined as follows*

- $E = [ir(D)]_{\leftrightarrow^*}$;
- $e \# e'$ *if there is no* $d \in \mathsf{K}(D)$ *such that* $e, e' \in [ir(d)]_{\leftrightarrow^*}$;
- $X \vdash e$ *if there exists* $i \in e$ *such that* $[ir(i) \setminus \{i\}]_{\leftrightarrow^*} \subseteq X$.

Given a morphism $f : D_1 \to D_2$, *its image* $\mathcal{E}(f) : \mathcal{E}(D_1) \to \mathcal{E}(D_2)$ *is defined for* $[i_1]_{\leftrightarrow^*} \in E$ *as* $\mathcal{E}(f)([i_1]_{\leftrightarrow^*}) = [i_2]_{\leftrightarrow^*}$, *where* $i_2 \in \delta(f(i_1), f(p(i_1)))$ *is minimal in the set, and* $\mathcal{E}(f)([i_1]_{\leftrightarrow^*})$ *is undefined if* $f(p(i_1)) = f(i_1)$.

The definition above is well-given: in particular, there is no ambiguity in the definition of the image of a morphism, since it can be shown that for all $i_2, i_2' \in \delta(f(i_1), f(p(i_1)))$ minimal, it holds $i_2 \leftrightarrow i_2'$.

Since in a prime domain irreducibles coincide with primes (Proposition 15), $\leftrightarrow$ is the identity (Lemma 20) and $\delta(d', d)$ is a singleton when $d \prec d'$, the construction above produces the prime ES $\mathsf{pES}(D)$ as defined in Section II.

Given a weak prime domain D, the finite configurations of the ES $\mathcal{E}(D)$ exactly correspond to the elements in $\mathsf{K}(D)$. Moreover, in such ES we have a minimal enabling $C \vdash_0 e$ when there is an irreducible in e (recall that events are equivalence classes of irreducibles) such that C contains all and only (the equivalence classes of) its predecessors.

Lemma 33 (compacts vs. finite configurations) *Let D be a weak prime domain and $C \subseteq \mathcal{E}(D)$ a set of events. Then C is a finite configuration in the* ES $\mathcal{E}(D)$ *iff there exists a (unique) $d \in \mathsf{K}(D)$ such that $C = [ir(d)]_{\leftrightarrow^*}$. Moreover, for any $e \in \mathcal{E}(D)$ we have that $C \vdash_0 e$ iff $C = [ir(i) \setminus \{i\}]_{\leftrightarrow^*}$ for some $i \in e$.*

Given the lemma above, it is now possible to state how weak prime domains relate to connected ES.

Proposition 34 *Let D be a weak prime domain. Then $\mathcal{E}(D)$ is a connected* ES.

At a categorical level, the constructions taking a weak prime domain to an ES and an ES to a domain (the domain of its configurations) establish a coreflection between the corresponding categories. This becomes an equivalence when it is restricted to the full subcategory of connected ES.

Theorem 35 (coreflection of ES and wDom) *The functors* $\mathcal{D} : \mathsf{ES} \to \mathsf{wDom}$ *and* $\mathcal{E} : \mathsf{wDom} \to \mathsf{ES}$ *form a coreflection. It restricts to an equivalence between* wDom *and* cES.

IV. Domain and event structure semantics for graph rewriting

In this section we consider graph rewriting systems where rules are left-linear but possibly not right-linear and thus, as an effect of a rewriting step, some items can be merged. We argue that weak prime domains and connected ES are the right tool for providing a concurrent semantics to this class of rewriting systems. More precisely, we show that the domain associated with a graph rewriting system by a generalisation of a classical construction is a weak prime domain and vice versa that each connected ES and thus each weak prime domain arise as the semantics of some graph rewriting system. In Subsection IV-A we review some background material and then in Subsections IV-B and IV-C we present our results.

A. Graph rewriting and concatenable traces

We open this section by reviewing the basic definitions about graph rewriting in the double-pushout approach [19]. We recall graph grammars and then introduce a notion of trace, which provides a representation of a sequence of rewriting steps that abstracts from the order of independent rewrites. Traces are then turned into a category $\mathsf{Tr}(\mathcal{G})$ of concatenable derivation traces [22].

Definition 36 *A* (directed) graph *is a tuple* $G = \langle N, E, s, t \rangle$, *where N and E are sets of nodes and edges, and $s, t : E \to N$ are the source and target functions. The components of a graph G are often denoted by N_G, E_G, s_G, t_G. A* graph morphism $f : G \to H$ *is a pair of functions* $f = \langle f_N : N_G \to N_H, f_E : E_G \to E_H \rangle$ *such that* $f_N \circ s = s' \circ f_E$ *and* $f_N \circ t = t' \circ f_E$. *We denote by* Graph *the category of graphs and graph morphisms*

An *abstract graph* $[G]$ is an isomorphism class of graphs. We work with typed graphs, i.e., graphs which are "labelled" over some fixed graph. Formally, given a graph T, the category of *graphs typed over* T, as introduced in [23], is the slice category $(\mathsf{Graph} \downarrow T)$, also denoted Graph_T.

Definition 37 (graph grammar) *A (T-typed graph)* rule *is a span* $(L \xleftarrow{l} K \xrightarrow{r} R)$ *in* Graph_T *where l is mono and not epi. The typed graphs L, K, and R are called the* left-hand side, *the* interface, *and the* right-hand side *of the rule, respectively. A (T-typed)* graph grammar *is a tuple* $\mathcal{G} = \langle T, G_s, P, \pi \rangle$, *where G_s is the* start (typed) graph, *P is a set of* rule names, *and π maps each rule name in P into a rule.*

Sometimes we write $p : (L \xleftarrow{l} K \xrightarrow{r} R)$ for denoting the rule $\pi(p)$. When clear from the context we omit the word "typed" and the typing morphisms. Note that we consider only *consuming* grammars, i.e., grammars where for every rule $\pi(p)$ the morphism l is not epi. This corresponds to the requirement on non-empty preconditions for Petri nets. Also note that rules are, by default, *left-linear*, i.e., morphism l is mono. If also morphism r is mono, the rule is called *right-linear*.

An example of graph grammar has been discussed in the introduction (see Fig. 2a). The type graph was left implicit: it can be found in the top part of Fig. 5. The typing morphisms for the start graph and the rules are implicitly represented by the labelling. Also observe that for the rules only the left-hand side L and the right-hand side R were reported. The same rules with the interface graph explicitly represented are in Fig. 5.

Definition 38 (direct derivation) *Let G be a typed graph, let* $p : (L \xleftarrow{l} K \xrightarrow{r} R)$ *be a rule, and let m^L be a* match, *i.e., a typed graph morphism* $m^L : L \to G$. *A* direct derivation δ *from G to H via p (based on m^L) is a diagram as in Fig. 6, where both squares are pushouts in* Graph_T. *We write* $\delta : G \xRightarrow{p/m} H$, *where* $m = \langle m^L, m^K, m^R \rangle$, *or simply* $\delta : G \xRightarrow{p} H$.

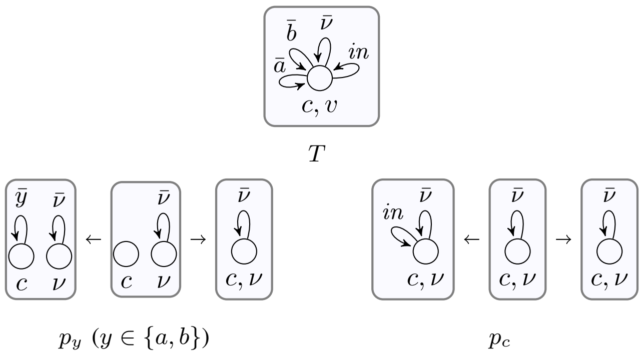


Fig. 5: The type graph and the rules of the grammar in Fig. 2a.

$$\begin{tikzcd} L \arrow[d, "m^L"'] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, "m^K"] & R \arrow[d, "m^R"] \\ G & D \arrow[l, "l^*"] \arrow[r, "r^*"'] & H \end{tikzcd}$$

Fig. 6: A direct derivation.

Since pushouts are defined only up to isomorphism, given isomorphisms $\kappa : G' \to G$ and $\nu : H \to H'$, also $G' \overset{p/m'}{\Longrightarrow} H$ with $m' = \langle \kappa^{-1} \circ m^L, m^K, m^R \rangle$ and $G \overset{p/m''}{\Longrightarrow} H'$ with $m'' = \langle m^L, m^K, \nu \circ m^R \rangle$ are direct derivations, denoted by $\kappa \cdot \delta$ and $\delta \cdot \nu$, respectively. Informally, the rewriting step removes (the image of) the left-hand side from G and replaces it by (the image of) the right-hand side R. The interface K (the common part of L and R) specifies what is preserved. For example, the transitions in Fig. 2b are all direct derivations. When rules are not right-linear, some items in the (image of the) interface are merged. This happens, e.g., for p_a and p_b.

Definition 39 (derivations) *Let $\mathcal{G} = \langle T, G_s, P, \pi \rangle$ be a graph grammar. A* derivation *$\rho : G_0 \Longrightarrow^*_{\mathcal{G}} G_n$ over $\mathcal{G}$ is a (possibly empty) sequence of direct derivations $\{G_{i-1} \overset{p_{i-1}}{\Longrightarrow} G_i\}_{i \in [1,n]}$ where $p_i \in P$ for $i \in [1, n]$. The graphs G_0 and G_n are called the* source *and the* target *of ρ, and denoted by $\mathsf{s}(\rho)$ and $\mathsf{t}(\rho)$, respectively. The* length *of ρ is $|\rho| = n$. Given two derivations ρ and ρ' such that $\mathsf{t}(\rho) = \mathsf{s}(\rho')$, their sequential composition $\rho\,;\,\rho' : \mathsf{s}(\rho) \Longrightarrow^* \mathsf{t}(\rho')$ is defined in the obvious way.*

When irrelevant/clear from the context, the subscript $\mathcal{G}$ is omitted. If $\rho : G \Longrightarrow^* H$ is a derivation, $|\rho| > 0$ and $\kappa : G' \to G$, $\nu : H \to H'$ are graph isomorphisms, then $\kappa \cdot \rho : G' \Longrightarrow^* H$ and $\rho \cdot \nu : G \Longrightarrow^* H'$ are defined as expected.

In the double pushout approach to graph rewriting, it is natural to consider graphs and derivations up to isomorphism. However some structure must be imposed on the start and end graph of an abstract derivation in order to have a meaningful notion of sequential composition. In fact, isomorphic graphs are, in general, related by several isomorphisms, while in order to concatenate derivations keeping track of the flow of causality one must specify how the items of the source and target isomorphic graphs should be identified. We follow [2], inspired by the theory of Petri nets [24]: we choose for each class of isomorphic typed graphs a specific graph, named the *canonical graph*, and we decorate the source and target graphs of a derivation with a pair of isomorphisms from the corresponding canonical graphs to such graphs.

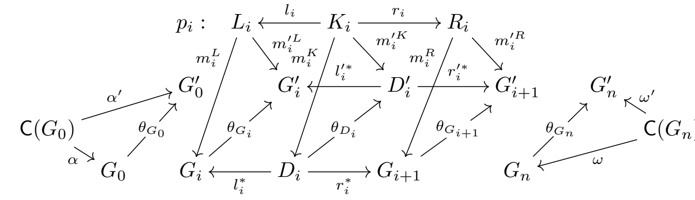


Fig. 7: Abstraction equivalence of decorated derivations.

Let C denote the operation that associates with each (T-typed) graph its *canonical graph*, thus satisfying $\mathsf{C}(G) \simeq G$ and if $G \simeq G'$ then $\mathsf{C}(G) = \mathsf{C}(G')$.

Definition 40 (decorated derivation) *A* decorated derivation *$\psi : G_0 \Longrightarrow^* G_n$ is a triple $\langle \alpha, \rho, \omega \rangle$, where $\rho : G_0 \Longrightarrow^* G_n$ is a derivation and $\alpha : \mathsf{C}(G_0) \to G_0$, $\omega : \mathsf{C}(G_n) \to G_n$ are isomorphisms. We define $\mathsf{s}(\psi) = \mathsf{C}(\mathsf{s}(\rho))$, $\mathsf{t}(\psi) = \mathsf{C}(\mathsf{t}(\rho))$ and $|\psi| = |\rho|$. The derivation is called* proper *if $|\psi| > 0$.*

Definition 41 (sequential composition) *Let $\psi = \langle \alpha, \rho, \omega \rangle$, $\psi' = \langle \alpha', \rho', \omega' \rangle$ be decorated derivations such that $\mathsf{t}(\psi) = \mathsf{s}(\psi')$. Their* sequential composition *$\psi; \psi'$ is defined, if ψ and ψ' are proper, as $\langle \alpha, (\rho \cdot \omega^{-1}); (\alpha' \cdot \rho'), \omega' \rangle$. Otherwise, if $|\psi| = 0$ then $\psi; \psi' = \langle \alpha' \circ \omega^{-1} \circ \alpha, \rho', \omega' \rangle$, and similarly, if $|\psi'| = 0$ then $\psi; \psi' = \langle \alpha, \rho, \omega \circ \alpha'^{-1} \circ \omega' \rangle$.*

We next define an abstraction equivalence that identifies derivations that differ only in representation details.

Definition 42 (abstraction equivalence) *Let $\psi = \langle \alpha, \rho, \omega \rangle$, $\psi' = \langle \alpha', \rho', \omega' \rangle$ be decorated derivations with $\rho : G_0 \Longrightarrow^* G_n$ and $\rho' : G'_0 \Longrightarrow^* G'_{n'}$ (whose i^{th} step is depicted in the lower rows of Fig. 7). They are* abstraction equivalent, *written $\psi \equiv^a \psi'$, if $n = n'$, $p_{i-1} = p'_{i-1}$ for all $i \in [1, n]$, and there exists a family of isomorphisms $\{\theta_{X_i} : X_i \to X'_i \mid X \in \{G, D\},\ i \in [1, n]\} \cup \{\theta_{G_0}\}$ between corresponding graphs in the two derivations such that (1) the isomorphisms relating the source and target commute with the decorations, i.e., $\theta_{G_0} \circ \alpha = \alpha'$ and $\theta_{G_n} \circ \omega = \omega'$; and (2) the resulting diagram (whose i^{th} step is represented in Fig. 7) commutes.*

Equivalence classes of decorated derivations with respect to $\equiv^a$ are called *abstract derivations* and denoted by $[\psi]_a$, where ψ is an element of the class.

From a concurrent perspective, derivations that only differ for the order in which two independent direct derivations are applied should not be distinguished. This is formalised by the classical shift equivalence on derivations.

Definition 43 (sequential independence) *Consider a derivation $G \overset{p_1/m_1}{\Longrightarrow} H \overset{p_2/m_2}{\Longrightarrow} M$ as in Fig. 8. Then, its components are* sequentially independent *if there exists an* independence pair *among them, i.e., two graph morphisms $i_1 : R_1 \to D_2$ and $i_2 : L_2 \to D_1$ such that $l_2^* \circ i_1 = m_{R_1}$, $r_1^* \circ i_2 = m_{L_2}$.*

$$
\begin{tikzcd}
L_1 \arrow[d, "m_{L_1}"'] & K_1 \arrow[l, "l_1"'] \arrow[r, "r_1"] \arrow[d, "m_{K_1}"] & R_1 \arrow[dr, "m_{R_1}"] \arrow[drrr, dotted, "i_1"'] & & L_2 \arrow[dl, "m_{L_2}"'] \arrow[dlll, dotted, "i_2"] & K_2 \arrow[l, "l_2"'] \arrow[r, "r_2"] \arrow[d, "m_{K_2}"] & R_2 \arrow[d, "m_{R_2}"] \\
G & D_1 \arrow[l, "l_1^*"] \arrow[rr, "r_1^*"'] & & H & & D_2 \arrow[ll, "l_2^*"] \arrow[r, "r_2^*"'] & M
\end{tikzcd}
$$

Fig. 8: Sequential independence for $\rho = G \overset{p_1/m_1}{\Longrightarrow} H \overset{p_2/m_2}{\Longrightarrow} M$.

Proposition 44 (interchange operator) *Let $\rho = G \overset{p_1/m_1}{\Longrightarrow} H \overset{p_2/m_2}{\Longrightarrow} M$ be a derivation whose components are sequentially independent via an independence pair ξ. Then, a derivation $IC_\xi(\rho) = G \overset{p_2/m_2^*}{\Longrightarrow} H^* \overset{p_1/m_1^*}{\Longrightarrow} M$ can be uniquely chosen, such that its components are sequentially independent via a canonical independence pair ξ^*.*

The interchange operator can be used to formalise a notion of shift equivalence [13], identifying (as for the analogous permutation equivalence of λ-calculus) those derivations which differ only for the scheduling of independent steps.

Definition 45 (shift equivalence) *The derivations ρ and ρ' are* shift equivalent*, written $\rho \equiv^{sh} \rho'$, if ρ' can be obtained from ρ by repeatedly applying the interchange operator.*

If we are interested in the way ρ' is obtained from ρ, we write $\rho \equiv^{sh}_{\sigma} \rho'$, for $\sigma : [1, n] \to [1, n]$ the associated permutation.

For instance, in Fig. 2b it is easy to see that the derivation $G_s \overset{p_a}{\Longrightarrow} G_b \overset{p_b}{\Longrightarrow} G_{ab}$ consists of sequentially independent direct derivations. It is shift equivalent to $G_s \overset{p_b}{\Longrightarrow} G_a \overset{p_a}{\Longrightarrow} G_{ab}$.

Two decorated derivations are said to be shift equivalent when the underlying derivations are, i.e., $\langle \alpha, \rho, \omega \rangle \equiv^{sh} \langle \alpha, \rho', \omega \rangle$ if $\rho \equiv^{sh} \rho'$. Then the equivalence of interest arises by joining abstraction and shift equivalence.

Definition 46 (concatenable traces) *We denote by $\equiv^c$ the equivalence on decorated derivations arising as the transitive closure of the union of the relations $\equiv^a$ and $\equiv^{sh}$. Equivalence classes of decorated derivations with respect to $\equiv^c$ are denoted as $[\psi]_c$ and are called* concatenable (derivation) traces.

We will sometimes annotate $\equiv^c$ with the permutation relating the equivalent permutations. Formally, $\equiv^c_\sigma$ can be defined inductively as: if $\psi \equiv^a \psi'$ then $\psi \equiv^c_{id} \psi'$, if $\psi \equiv^{sh}_\sigma \psi'$ then $\psi \equiv^c_\sigma \psi'$, and if $\psi \equiv^c_\sigma \psi'$ and $\psi' \equiv^c_{\sigma'} \psi''$ then $\psi \equiv^c_{\sigma' \circ \sigma} \psi''$.

The sequential composition of decorated derivations lifts to composition of derivation traces so that we can consider the corresponding category.

Definition 47 (category of concatenable traces) *Let $\mathcal{G}$ be a graph grammar. The* category of concatenable traces *of $\mathcal{G}$, denoted by $\mathsf{Tr}(\mathcal{G})$, has abstract graphs as objects and concatenable traces as arrows.*

B. A weak prime domain for a grammar

For a grammar $\mathcal{G}$ we obtain a partially ordered representation of its derivations starting from the initial graph by considering the concatenable traces ordered by prefix. Formally, as done in [2], [3] for linear grammars, we consider the category $([G_s] \downarrow \mathsf{Tr}(\mathcal{G}))$, which, by definition of sequential composition between traces, is easily shown to be a preorder.

Proposition 48 *Let $\mathcal{G}$ be a graph grammar. Then the category $([G_s] \downarrow \mathsf{Tr}(\mathcal{G}))$ is a preorder.*

Explicitly, elements of the preorder are concatenable traces $[\psi]_c : [G_s] \to [G]$ and, for $[\psi']_c : [G_s] \to [G']$, we have $[\psi]_c \sqsubseteq [\psi']_c$ if there is $\psi'' : G \to G'$ such that $\psi; \psi'' \equiv^c \psi'$. Therefore, given two concatenable traces $[\psi]_c : [G_s] \to [G]$ and $[\psi']_c : [G_s] \to [G']$, if $[\psi]_c \sqsubseteq [\psi']_c \sqsubseteq [\psi]_c$ then ψ can be obtained from ψ' by composing it with a zero-length trace. Hence the elements of the partial order induced by $([G_s] \downarrow \mathsf{Tr}(\mathcal{G}))$ intuitively consist of classes of concatenable traces whose decorated derivations are related by an isomorphism that has to be consistent with the decoration of the source. Once applied to the grammar in Fig. 2a, this construction produces a domain isomorphic to that in Fig. 2c.

Lemma 49 *Let $\mathcal{G}$ be a graph grammar. The partial order induced by $([G_s] \downarrow \mathsf{Tr}(\mathcal{G}))$, denoted $\mathcal{P}(\mathcal{G})$, has as elements $\langle \psi \rangle_c = \{[\psi \cdot \nu]_c \mid \nu : \mathsf{t}(\psi) \overset{\sim}{\to} \mathsf{t}(\psi)\}$ and $\langle \psi \rangle_c \sqsubseteq \langle \psi' \rangle_c$ if $\psi; \psi'' \equiv^c \psi'$ for some decorated derivation ψ''.*

The domain of interest is then obtained by ideal completion of $\mathcal{P}(\mathcal{G})$, with (the principal ideals generated by) the elements in $\mathcal{P}(\mathcal{G})$ as compact elements. In order to give a proof for this, we need a preliminary technical lemma that essentially proves the existence and provides the shape of the least upper bounds in the domain of traces.

Lemma 50 (properties of $\equiv^c$) *1) Let ψ, ψ' be decorated derivations, and ψ_1, ψ'_1 such that $\psi; \psi_1 \equiv^c_\sigma \psi'; \psi'_1$ and $n = |\{j \in [|\psi|, |\psi; \psi_1| - 1] \mid \sigma(j) < |\psi'|\}|$. Then for all ϕ_2, ϕ'_2 such that $\psi; \phi_2 \equiv^c \psi'; \phi'_2$ it holds $|\phi_2| \geq n$ and there are ψ_2, ψ'_2, ψ_3 such that*

- *$\psi; \psi_1 \equiv^c \psi; \psi_2; \psi_3$*
- *$\psi; \psi_2 \equiv^c \psi'; \psi'_2$*
- *$|\psi_2| = n$*

2) Let ψ, ψ' be decorated derivations and ψ_1, ψ'_1, ψ_2, ψ'_2 such that $\psi; \psi_1 \equiv^c_{\sigma_1} \psi'; \psi'_1$ and $\psi; \psi_2 \equiv^c_{\sigma_2} \psi'; \psi'_2$ with ψ_1, ψ_2 of minimal length. Then $\psi_1 \equiv^c_\sigma \psi_2 \cdot \nu$, where $\nu : \mathsf{t}(\psi_2) \to \mathsf{t}(\psi_2)$ is some graph isomorphism and $\sigma(j) = \sigma_2^{-1}(\sigma_1(j + |\psi|)) - |\psi|$ for $j \in [0, |\psi_1| - 1]$.

Relying on the results above we can easily prove that the ideal completion of the partial order of traces is a domain.

Proposition 51 (domain of traces) *Let $\mathcal{G}$ be a graph grammar. Then $\mathcal{D}(\mathcal{G}) = \mathsf{Idl}(\mathcal{P}(\mathcal{G}))$ is a domain.*

We can show that $\mathcal{D}(\mathcal{G})$ is a weak prime domain. The proof relies on the fact that irreducibles are (the principal ideals of) elements of the form $\langle \epsilon \rangle_c$, where $\epsilon = \psi; \delta$ is a decorated derivation such that its last direct derivation δ cannot be switched back, i.e., minimal traces enabling some direct derivation. These are called *pre-events* in [2], [3], where graph grammars are linear and thus, consistently with Lemma 15, such elements provide the primes of the domain.

Two irreducibles $\langle \epsilon \rangle_c$ and $\langle \epsilon' \rangle_c$ are interchangeable when they are different minimal traces for the same direct derivation.

Theorem 52 (weak prime domains from graph grammars) *Let $\mathcal{G}$ be a graph grammar. Then $\mathcal{D}(\mathcal{G})$ is a weak prime domain.*

Note that when the rules are right-linear the domain and ES semantics specialises to the usual prime event structure semantics (see [2]–[4]), since the construction of the domain in the present paper is formally the same as in [2].

C. Any connected ES *is generated by some grammar*

By Theorem 52, given a graph grammar $\mathcal{G}$ the domain $\mathcal{D}(\mathcal{G})$ is weak prime. We next show that also the converse holds, i.e., any connected ES (and thus any weak prime domain) is generated by a suitable graph grammar. This shows that weak prime domains and connected ES are precisely what is needed to capture the concurrent semantics of non-linear graph grammars, and thus strengthen our claim that they represent the right structure for modelling formalisms with fusions.

Construction (graph grammar for a connected ES) Let $\langle E, \#, \vdash \rangle$ be a connected ES. The grammar $\mathcal{G}_E = \langle T, P, \pi, G_s \rangle$ is defined as follows.

First, for every element $e \in E$, we define the following graphs, which are then used as basic building blocks

- I_e and S_e, as shown in Fig. 9(a) and Fig. 9(b);
- let U_e denote the set-theoretical product of the minimal enablings of e, i.e., $U_e = \Pi\{X \subseteq E \mid X \vdash_0 e\}$; for every tuple $u \in U_e$ we define the graph $L_{u,e}$ as in Fig. 9(c).

Moreover, for every pair of events $e, e' \in E$ such that $e \# e'$, we define a graph $C_{e,e'}$ as in Fig. 9(d).

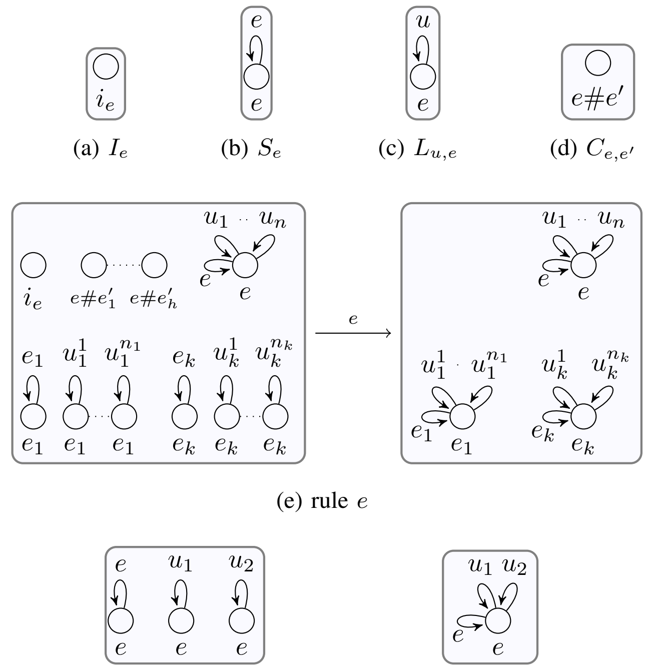
The set of productions is $P = E$, i.e., we add a rule for every event $e \in E$, and we define such rule in a way that

- it deletes I_e and $C_{e,e'}$ for each $e' \in E$ such that $e \# e'$;
- it preserves the graph $S_e \cup \bigcup_{u \in U_e} L_{u,e}$;
- for all $e' \in E$, for all graphs $L_{u,e'}$ such that e occurs in u, it merges the corresponding nodes and that of $S_{e'}$ into one.

The graph $S_e \cup \bigcup_{u \in U_e} L_{u,e}$ arises from S_e and $L_{u,e}$, $u \in U_e$ by merging all the nodes (we use $\bigcup$ and $\biguplus$ to denote union and disjoint union, respectively, with a meaning illustrated in Figs. 9(f) and (g).) Hence, there is a match for the rule e only if S_e and all $L_{u,e}$ for $u \in U_e$ have been merged and this happens if and only if at least one minimal enabling of e has been entirely executed. The deletion of the graphs $C_{e,e'}$ establishes the needed conflicts. The rule is consuming since it deletes the node of graph I_e. The rule is schematised in Fig. 9(e), where it is intended that e occurs in $u_j^1, \ldots, u_j^{n_j}$ for $u_j^i \in U_{e_j}$, $j \in [1, k], i \in [1, n_k]$. Moreover $e'_1, \ldots, e'_h$ are the events in conflict with e and, finally, $U_e = \{u_1, \ldots, u_n\}$.

The start graph is just the disjoint union of all the basic graphs introduced above

$$G_s = (\bigcup_{e \# e'} C_{e,e'}) \cup \bigcup_{e \in E} (I_e \cup S_e \uplus \biguplus_{u \in U_e} L_{u,e})$$



(a) I_e (b) S_e (c) $L_{u,e}$ (d) $C_{e,e'}$

(e) rule e

(f) $S_e \uplus L_{u_1,e} \uplus L_{u_2,e}$ (g) $S_e \cup L_{u_1,e} \cup L_{u_2,e}$

Fig. 9: Some graphs illustrating the construction of $\mathcal{G}_E$.

For space limitations the interfaces of the rules are not given explicitly. They can be deduced from the left and right-hand side, and the labelling. The same applies to the type graph.

It is not difficult to show that the grammar $\mathcal{G}_E$ generates exactly the ES E.

Theorem 53 *Let $\langle E, \#, \vdash \rangle$ be a connected* ES*. Then, E and $\mathcal{E}(\mathcal{D}(\mathcal{G}_E))$ are isomorphic connected* ES*.*

Example 54 *Consider the running example* ES*, from Example 6, with set of events $\{a, b, c\}$, empty conflict relation and minimal enablings $\{a\} \vdash_0 c$ and $\{b\} \vdash_0 c$. The associated grammar is depicted in Fig. 10.*

As a further example, consider an ES *E_1 with events $\{a, b, c, d, e\}$. The conflict relation $\#$ is given by $e \# d$ and the minimal enablings by $\emptyset \vdash_0 a$, $\emptyset \vdash_0 b$, $\emptyset \vdash_0 c$, $\emptyset \vdash_0 e$, $\{a, b\} \vdash_0 d$, and $\{c\} \vdash_0 d$. The grammar is in Fig. 11.*

V. Conclusions and related work

In the paper we provided a characterisation of a class of domains, referred to as weak prime algebraic domains, appropriate for describing the concurrent semantics of those formalisms where a computational step can merge parts of the state. We established a categorical equivalence between weak prime algebraic domains and a suitably defined class of connected event structures. We also proved that the category of general unstable event structures coreflects into the category of weak prime algebraic domains. The appropriateness of the class of weak prime domains is witnessed by the results in the second part of the paper that show that weak prime algebraic domains are precisely those arising from left-linear graph

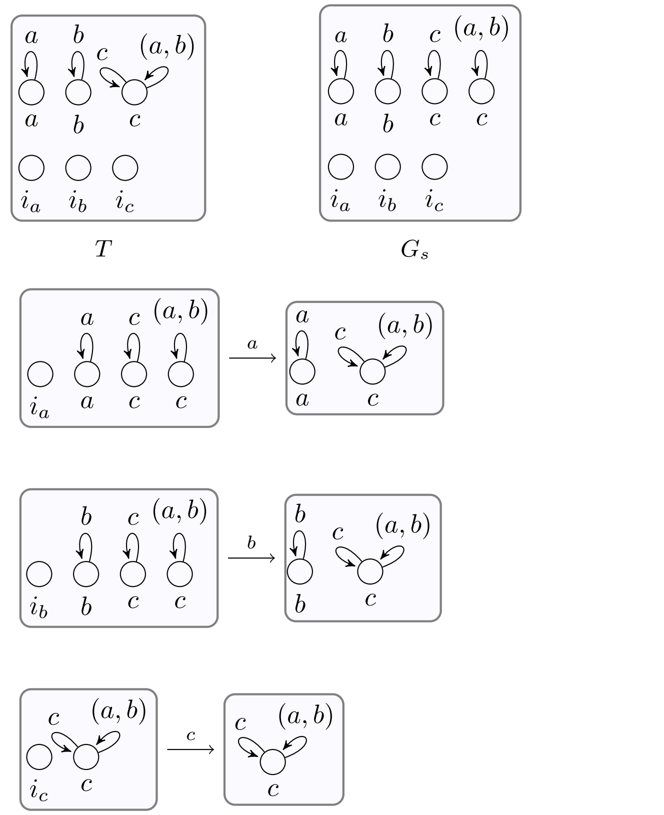


Fig. 10: The grammar associated with our running example.

rewriting systems, i.e., those systems where rules besides generating and deleting can also merge graph items.

Technically, the starting point is the relaxation of the stability condition for event structures. As already noted by Winskel in [5] "[t]he stability axiom would go if one wished to model processes which had an event which could be caused in several compatible ways [...]; then I expect complete irreducibles would play a similar role to complete primes here". Indeed, the correspondence between irreducibles and weak primes, based on the notion of interchangeability, is the ingenious step that allowed us to obtain a smooth extension of the classical duality between prime event structures and prime algebraic domains.

The coreflection between the category of general unstable event structures (with binary conflict) and the one of weak prime algebraic domains says that the latter are exactly the partial orders of configurations of the former. Such class of domains has been studied originally in [20] where, generalising the work on concrete domains and sequentiality [25], a characterisation is given in terms of a set of axioms expressing properties of prime intervals. A similar characterisation for event structures with non-binary conflict is in [21]. We consider our simple characterisation of this class of domains, where weak primes intuitively account for events in a computation, as a valuable contribution of the paper. We plan to provide an in depth comparison with these previous results in the full version of the paper. In brief, a formal bridge between the two characterisations can be established by observing that, roughly speaking, weak primes correspond to executions of events with their minimal enablings, while intervals can be seen as executions of events in a generic configuration.

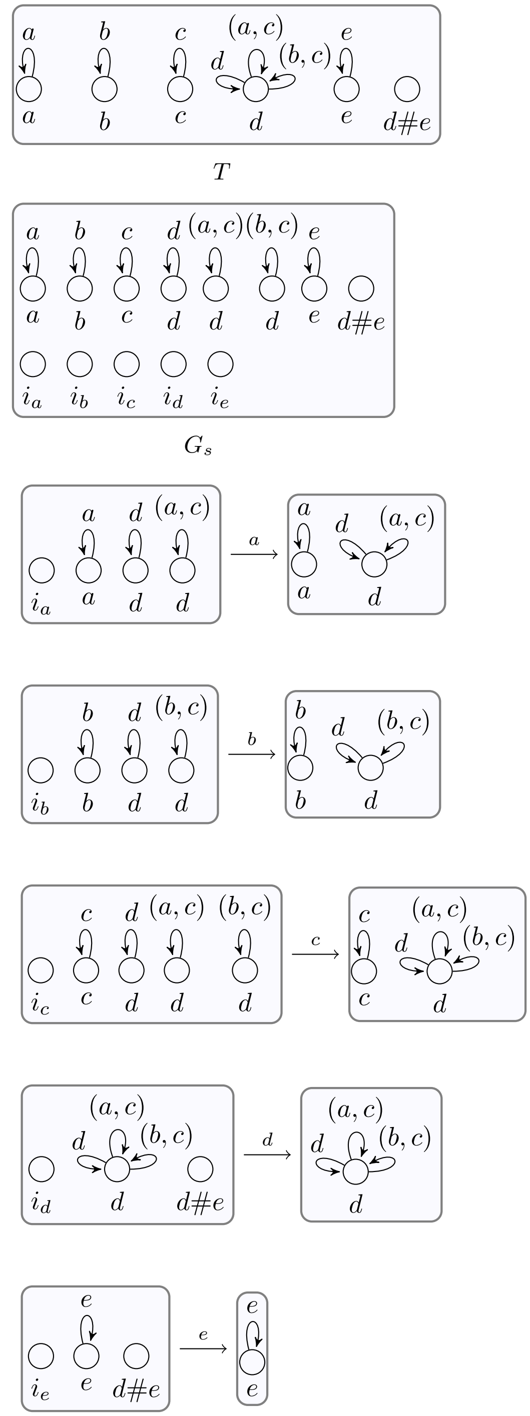


Fig. 11: The grammar for the ES in example 54.

The paper [26] studies a characterisation of the partial order of configurations for a variety of classes of event structures in terms of axiomatisability of the associated propositional theories. Even if the focus is here mainly on event structures that generalise Winskel's ones, we believe that our work can provide interesting suggestions for further development.

The need of resorting to unstable event structures for modelling the concurrent computations of name passing process calculi has been observed by several authors. In particular, in [16] an event structure semantics for the pi-calculus is defined by relying on structures that are tailored for parallel extrusions. These are labelled unstable event structures with the constraint that two minimal enablings can differ only for one event (intuitively, the extruder). The corresponding domain of configurations is weak prime algebraic but the ES fails to be connected since non-connected minimal enablings are admitted (roughly, because identical events in disconnected minimal enablings are identified via the labelling).

We finally remark that a possibility for recovering a notion of causality based on prime event structures also for rule-based formalisms with fusions is to introduce suitable restrictions on the concurrent applicability of rules. Indeed, the lack of stability seems to arise essentially from considering as concurrent those fusions that act on common items. Preventing fusions to act on already merged items, one may lose some concurrency, yet gaining a definite notion of causality. Technically, a prime event structure can be obtained for left-linear grammars by restricting the applicability condition: the match must be such that the pair $\langle l; m^L, r\rangle$ of Fig. 6 is jointly mono. This essentially means that those items that have been already fused, should not be fused again. This is indeed the intuition behind the proposal advanced in [27]. Concerning our running example, this requirement would forbid the reachability of graphs G_{ab} and G_c in Fig. 2(b), and in turn this would imply that the domain of configurations is the one depicted in Fig. 4, with the limits concerning expressiveness and event identity that we already remarked there.

Acknowledgements: We are grateful to the anonymous referees for their insightful comments and suggestions on the submitted version of this paper.

References

[1] M. Nielsen, G. Plotkin, and G. Winskel, "Petri Nets, Event Structures and Domains, Part 1," *Theoretical Computer Science*, vol. 13, pp. 85–108, 1981.

[2] P. Baldan, A. Corradini, H. Ehrig, M. Löwe, U. Montanari, and F. Rossi, "Concurrent semantics of algebraic graph transformation systems," in *Handbook of Graph Grammars and Computing by Graph Transformation*, G. Rozenberg, Ed. World Scientific, 1999, vol. III: Concurrency, pp. 107–187.

[3] P. Baldan, "Modelling concurrent computations: from contextual Petri nets to graph grammars," Ph.D. dissertation, University of Pisa, 2000.

[4] G. Schied, "On relating rewriting systems and graph grammars to event structures," in *Dagstuhl Seminar 9301 on Graph Transformations in Computer Science*, ser. LNCS, H.-J. Schneider and H. Ehrig, Eds., vol. 776. Springer, 1994, pp. 326–340.

[5] G. Winskel, "Event structure semantics for CCS and related languages," University of Aarhus, Tech. Rep. DAIMI PB-159, 1983.

[6] D. Varacca and N. Yoshida, "Typed event structures and the linear pi-calculus," *Theoretical Computer Science*, vol. 411, no. 19, pp. 1949–1973, 2010.

[7] R. Bruni, H. C. Melgratti, and U. Montanari, "Event structure semantics for nominal calculi," in *CONCUR 2006*, ser. LNCS, C. Baier and H. Hermanns, Eds., vol. 4137. Springer, 2006, pp. 295–309.

[8] G. Winskel, "Events, causality and symmetry," *Computer Journal*, vol. 54, no. 1, pp. 42–57, 2011.

[9] J. Pichon-Pharabod and P. Sewell, "A concurrency semantics for relaxed atomics that permits optimisation and avoids thin-air executions," in *POPL 2016*, R. Bodík and R. Majumdar, Eds. ACM, 2016, pp. 622–633.

[10] A. Jeffrey and J. Riely, "On thin air reads towards an event structures model of relaxed memory," in *LICS 2016*, M. Grohe, E. Koskinen, and N. Shankar, Eds. ACM, 2016, pp. 759–767.

[11] M. Dumas and L. García-Bañuelos, "Process mining reloaded: Event structures as a unified representation of process models and event logs," in *Petri Nets 2015*, ser. LNCS, R. R. Devillers and A. Valmari, Eds., vol. 9115. Springer, 2015, pp. 33–48.

[12] J. Meseguer, "Conditional rewriting logic as a unified model of concurrency," *Theoretical Computer Science*, vol. 96, no. 1, pp. 73–155, 1992.

[13] A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe, "Algebraic approaches to graph transformation I: Basic concepts and double pushout approach," in *Handbook of Graph Grammars and Computing by Graph Transformation. Volume 1: Foundations.*, G. Rozenberg, Ed. World Scientific, 1997.

[14] J. Lévy, "Optimal reductions in the lambda-calculus," in *To H.B. Curry, Essays on Combinatory Logic, Lambda Calculus and Formalism*, J. Seldin and J. Hindley, Eds. Academic Press, 1980, pp. 159–191.

[15] G. Winskel, "Event Structures," in *Petri Nets: Central Models and Their Properties*, ser. LNCS, W. Brauer, W. Reisig, and G. Rozenberg, Eds., vol. 255. Springer, 1987, pp. 325–392.

[16] S. Crafa, D. Varacca, and N. Yoshida, "Event structure semantics of parallel extrusion in the π-calculus," in *FoSSaCS 2012*, ser. LNCS, L. Birkedal, Ed., vol. 7213. Springer, 2012, pp. 225–239.

[17] F. Gadducci, "Graph rewriting and the π-calculus," *Mathematical Structures in Computer Science*, vol. 17, no. 3, pp. 1–31, 2007.

[18] I. Phillips, I. Ulidowski, and S. Yuen, "Modelling of bonding with processes and events," in *RC 2013*, ser. LNCS, G. W. Dueck and D. M. Miller, Eds., vol. 7948. Springer, 2013, pp. 141–154.

[19] H. Ehrig, "Tutorial introduction to the algebraic approach of graph-grammars," in *TAGT 1986*, ser. LNCS, H. Ehrig, M. Nagl, G. Rozenberg, and A. Rosenfeld, Eds., vol. 291. Springer, 1987, pp. 3–14.

[20] G. Winskel, "Events in computation," Ph.D. dissertation, University of Edinburgh, 1980.

[21] M. Droste, "Event structures and domains," *Theoretical Computer Science*, vol. 68, no. 1, pp. 37–47, 1989.

[22] A. Corradini, H. Ehrig, M. Löwe, U. Montanari, and F. Rossi, "An event structure semantics for graph grammars with parallel productions," in *TAGT 1994*, ser. LNCS, J. Cuny, H. Ehrig, G. Engels, and G. Rozenberg, Eds., vol. 1073. Springer, 1996.

[23] A. Corradini, U. Montanari, and F. Rossi, "Graph processes," *Fundamenta Informaticae*, vol. 26, no. 3/4, pp. 241–265, 1996.

[24] P. Degano, J. Meseguer, and U. Montanari, "Axiomatizing the algebra of net computations and processes," *Acta Informatica*, vol. 33, no. 7, pp. 641–647, 1996.

[25] G. Kahn and G. Plotkin, "Concrete domains," *Theoretical Computer Science*, vol. 121, no. 1, pp. 187–277, 1993, based on [28].

[26] R. van Glabbeek and G. Plotkin, "Configuration structures, event structures and Petri nets," *Theoretical Computer Science*, vol. 410, no. 41, pp. 4111–4159, 2009.

[27] P. Baldan, F. Gadducci, and U. Montanari, "Concurrent rewriting for graphs with equivalences," in *CONCUR 2006*, ser. LNCS, C. Baier and H. Hermanns, Eds., vol. 4137. Springer, 2006, pp. 279–294.

[28] G. Kahn and G. Plotkin, "Domaines concretes," INRIA Paris, Tech. Rep. 336, 1978.