Categorical Connections between Local Event Structures and Local Traces

H.C.M. Kleijn[1], R. Morin[2], and B. Rozoy[2]

[1] LIACS, Leiden University, P.O. Box 9512, 2300 RA Leiden, The Netherlands
[2] L.R.I., Bât. 490, Université de Paris Sud, F-91405 Orsay Cedex, France

Abstract. Local event structures and local traces are generalizations of the classical prime event structures and Mazurkiewicz' traces in which independence is no longer a global binary property. We consider the problem of lifting the categorical connection between prime event structures and Mazurkiewicz' traces to this more general level. Using a generic approach it is shown how certain subcategories of local event structures and local trace languages can be related. Moreover, every coreflection between subcategories generalizing the connection between prime event structures and Mazurkiewicz' traces fits into this generic approach.

Introduction

The traces introduced by Mazurkiewicz [5] and prime event structures [6] are well-known abstract models for describing the behavior of concurrent systems, in particular 1-safe Petri nets. Whereas traces can be used to describe the non-conflicting sequential executions of the system together with an equivalence relation induced by an independence relation over the actions, a prime event structure provides explicit information on the relationships between events in terms of a partial ordering and a binary conflict relation. Despite the fact that the prime event structure model is more abstract than the trace model, they are closely related [8]. In particular, a coreflection between the categories of prime event structures and of Mazurkiewicz trace languages is established [10]. This categorical approach allows not only to compare the models as objects, but also to compare their behavioral aspects (see, e.g., [1,7]).

When an abstract model is meant to represent the behavior of dynamic systems, it is reasonable to extend it to a category by equipping it with behavior preserving morphisms to capture a notion of simulation. Categories can be compared by functors which relate objects to objects and morphisms to morphisms, and thus preserve the dynamic behavior of the systems. An adjunction between categories consists of two functors, one in each direction, that fit together in a particular way. This is a formal way to express that one model is more abstract than another, as it allows to canonically represent objects from the more concrete model that are mapped to the same object in the more abstract model. If in addition going from an object in the more abstract model to an object in the more concrete model and then back using the functors, leads to an isomorphic object, the adjunction is called a coreflection. Thus establishing a coreflection between two categories proves a very strong relationship, as it shows that the

functor to the abstract model gives a faithful description of the concrete model. For more technical details of categories, functors, adjunctions and coreflections, the reader is referred to [9].

Local traces and local event structures were introduced in [2,3] to lift the semantical theory of 1-safe Petri nets to the level of more general Petri nets in which concurrency and conflict are not structural properties, but depend on the current marking (the state of the system). Therefore a local independence relation describes which sets of actions may occur concurrently after a given execution of the system, while local event structures require a concurrency axiom local to their configurations. Both the local trace semantics and the local event structure semantics of Petri nets are respectively proper conservative extensions of the trace semantics and the prime event structure semantics of 1-safe Petri nets. These extended models have been considered in [4] as independent notions and their relationship has been investigated. As it is relatively easy to go from local event structures to local trace languages, the main issue in that paper is how to associate a local event structure to a local independence relation and in particular how to identify events. Following the classical approach, events are defined as equivalence classes of prime intervals. Two different equivalence relations for prime intervals are discussed: one, Projectivity, corresponds to the equivalence used to relate Mazurkiewicz' traces and prime event structures; the other, History, corresponds to the equivalence used in [3] to relate Petri nets to local event structures. The latter equivalence leads to a coreflection between main subcategories which extends the original coreflection between prime event structures and Mazurkiewicz trace languages.

This paper focuses on a generic approach towards the categorical relationship between the categories of local event structures and local trace languages. First, we recall the straitforward and intuitive representation of local event structures as local trace languages. This map easily extends to a functor which however admits *no* right-adjoint (Th. 2.4) except for restricted subcategories of each model [10,4]. Next we introduce a notion of *punctuation*, based on equivalences of prime intervals, which admits Projectivity and History as particular examples; our main result is that any punctuation determines a coreflection between some associated subcategories of local event structures and local trace languages (Th. 2.12). Moreover we show that any generalization of the coreflection between prime event structures and trace languages may be obtained by this generic coreflection (Th. 3.2). Due to the page limit, most proofs are totally omitted[1].

1 Basic Notions and Results

Preliminaries. We will use the following notations: for any (possibly infinite) alphabet Σ, and any words $u \in \Sigma^\star$, $v \in \Sigma^\star$, we write $u \leq v$ if u is a prefix of v, i.e. there is $z \in \Sigma^\star$ such that $u.z = v$; the empty word is denoted by ε. We write $|u|_a$ for the number of occurrences of $a \in \Sigma$ in $u \in \Sigma^\star$ and $\wp_f(\Sigma)$ denotes the

[1] A detailed study is available at `http://www.lri.fr/~morin/papers/kmr.ps`.

set of finite subsets of Σ; for any $p \in \wp_f(\Sigma)$, $\mathrm{Lin}(p) = \{u \in p^\star \mid \forall a \in p, |u|_a = 1\}$ is the set of linearisations of p. Finally, if $\lambda : \Sigma \rightharpoonup \Sigma'$ is a partial function from Σ to Σ', we also write $\lambda : \Sigma^\star \to \Sigma'^\star$ and $\lambda : \wp_f(\Sigma) \to \wp_f(\Sigma')$ to denote the naturally associated monoid morphisms.

Local Trace Languages. Local traces form a generalization of the classical Mazurkiewicz' traces by their being based on an independence relation which is left-context dependent and which specifies sets of independent actions rather than pairs.

DEFINITION 1.1. *A* local independence relation *on Σ is a non-empty subset I of $\Sigma^\star \times \wp_f(\Sigma)$. The* (local) trace equivalence $\sim$ *induced by I is the least equivalence on $\Sigma^\star$ such that*

TE_1*:* $\forall u, u' \in \Sigma^\star, \forall a \in \Sigma, u \sim u' \Rightarrow u.a \sim u'.a$*;*

TE_2*:* $\forall (u,p) \in I, \forall p' \subseteq p, \forall v_1, v_2 \in \mathrm{Lin}(p'), u.v_1 \sim u.v_2$*.*

A (local) trace *is an $\sim$-equivalence class $[u]$ of a word $u \in \Sigma^\star$.*

By TE_1 local trace equivalences are right-congruences. TE_2 asserts that for every subset of actions which are independent after a sequence u, all sequences obtained by executing first u and then in an arbitrary order the actions from this subset, are equivalent. Note also that local trace equivalences are Parikh equivalences.

A local independence relation can be thought of as a representation of the behavior of a concurrent system. It provides information on possible sequential observations as well as information on their equivalence. Thus every local independence relation defines a prefix-closed language of sequential observations and a set of traces, the equivalence classes of these sequential observations. As observed in [4], the assumptions in these definitions can be translated into explicit additional conditions on the local independence relation without affecting the resulting sets of observations and traces. A local independence relation satisfying these additional conditions is called complete and is a maximal representative among local independence relations defining the same sequential observations and traces. In this paper we directly define local trace languages as combinations of a language (of sequences) and a complete local independence relation.

DEFINITION 1.2. *A* local trace language *(LTL) over Σ is a structure $\mathcal{L} = (\Sigma, I, L)$ where $L \subseteq \Sigma^\star$ and I is a local independence relation on Σ such that*

LTL_1*:* $(u,p) \in I \wedge\ p' \subseteq p \Rightarrow (u,p') \in I$*;*

LTL_2*:* $(u,p) \in I \wedge\ p' \subseteq p \wedge\ v \in \mathrm{Lin}(p') \Rightarrow (u.v, p \setminus p') \in I$*;*

LTL_3*:* $u \sim u' \wedge\ (u,p) \in I \Rightarrow (u',p) \in I$*;*

LTL_4*:* $(u.a, \emptyset) \in I \Rightarrow (u, \{a\}) \in I$*;*

LTL_5*:* $u \in L \Leftrightarrow (u, \emptyset) \in I$*.*

LTL_1 through LTL_4 are the requirements which make the local independence relation complete. LTL_1 makes explicit what TE_2 from Def. 1.1 guarantees for the trace equivalence: if a set of actions p can be executed concurrently after u, then so can any subset of p; moreover, following LTL_2, the step p can be split into a sequential execution v and a concurrent step of the remaining actions. LTL_3 states that after two equivalent sequences the independency and thus unorderedness of

actions is the same; it corresponds to the right-congruence property TE_1 from Definition 1.1. LTL_4 guarantees that whenever $u.a$ is a sequential execution, then action a is allowed as a step after u.

LTL_5 ensures compatibility of L, the set of sequential observations, and the local independence relation I. From LTL_4 and LTL_1 it then follows that L is prefix-closed. By LTL_3 we know that L is closed under the trace equivalence induced by I. Thus L is precisely the set of sequential observations associated to I in [4].

Local Event Structures. A local event structure is a family of configurations equipped with an enabling relation that specifies locally the possible concurrency of events.

DEFINITION 1.3. *A* local event structure *(LES) is a triple* $\mathcal{E} = (E, C, \vdash)$ *where* E *is a set of events,* $C \subseteq \wp_f(E)$ *is a set of finite subsets of events called configurations and* $\vdash \subseteq C \times \wp_f(E)$ *is an enabling relation such that*

LES_1*:* $(\emptyset \vdash \emptyset) \wedge \ (\forall e \in E, \exists c \in C, e \in c)$*;*
LES_2*:* $\forall c \in C: c \neq \emptyset \Rightarrow \exists e \in c, c \setminus \{e\} \vdash \{e\}$*;*
LES_3*:* $\forall c \in C, \forall p \in \wp_f(E)$*:* $c \vdash p \Rightarrow c \cap p = \emptyset$*;*
LES_4*:* $\forall c \in C, \forall p \in \wp_f(E), \forall p' \subseteq p$*:* $c \vdash p \Rightarrow (c \vdash p' \wedge \ c \cup p' \vdash p \setminus p')$*.*

LES_1 guarantees that the empty set is always a configuration and that the enabling relation is never empty. Also by LES_1, each event occurs in at least one configuration. LES_2 ensures that every non-empty configuration can be reached from the (initial) empty configuration. LES_3 implies that each event occurs at most once and by LES_4 each concurrent set can be split arbitrarily into subsets of concurrent events.

With each local event structure $\mathcal{E}$ a set of (finite) sequential observations can be associated which we call the paths of $\mathcal{E}$; formally, $\mathrm{Paths}(\mathcal{E}) = \{e_1 ... e_n \in E^\star \mid \forall i \in [1, n], \{e_1, ..., e_{i-1}\} \vdash \{e_i\}\}$. As shown in [3], an event appears at most once along a path and each path u leads to a unique configuration $\mathrm{Cfg}(u)$ defined by $\mathrm{Cfg}(u) = \{e \mid |u|_e = 1\}$.

It is easy to associate with a local event structure $\mathcal{E}$ a local trace language $\mathfrak{tl}(\mathcal{E})$ with the same sequential observations and a local independence relation faithfully representing the concurrency in $\mathcal{E}$ (see also [4]).

DEFINITION 1.4. *Let* $\mathcal{E} = (E, C, \vdash)$ *be a local event structure. The local trace language* $\mathfrak{tl}(\mathcal{E})$ *associated to* $\mathcal{E}$ *is* $\mathfrak{tl}(\mathcal{E}) = (E, I, \mathrm{Paths}(\mathcal{E}))$ *where* $I = \{(u, p) \in \Sigma^\star \times \wp_f(\Sigma) \mid u \in \mathrm{Paths}(\mathcal{E})$ *and* $\mathrm{Cfg}(u) \vdash p\}$.

Equivalences of Prime Intervals. In order to associate a local event structure with a given local trace language $\mathcal{L} = (\Sigma, I, L)$ one has to define events. The event structure should properly reflect the sequential behavior and the independencies represented by the local trace language. Thus we want events to represent occurrences of actions. For the prefix-closed language L occurrences of actions can be defined as *prime intervals* which are pairs $(u, a) \in \Sigma^\star \times \Sigma$ such that $u.a \in L$; we write $\mathrm{Pr}(\mathcal{L})$ for the set of prime intervals of $\mathcal{L}$. It may however

be the case that different occurrences of an action (hence different prime intervals) should correspond to the same event: for instance, if $(u, \{a, b\}) \in I$ then actions a and b may occur simultaneously after u; an observer cannot distinguish the occurrence of a after u from the occurrence of a after $u.b$; thus (u, a) and $(u.b, a)$ must be identified as the same occurrence of a. Furthermore if u and u' are equivalent (in the same local trace), then the prime intervals (u, a) and (u', a) should not be distinguished either. For these reasons we need an equivalence relation over the prime intervals of $\mathcal{L}$ when associating a local event structure with $\mathcal{L}$. The requirements which any such equivalence should satisfy are defined now.

DEFINITION 1.5. *Let $\mathcal{L} = (\Sigma, I, L)$ be a local trace language; an* equivalence of prime intervals *of $\mathcal{L}$ is an equivalence $\asymp_{\mathcal{L}}$ over $\mathrm{Pr}(\mathcal{L})$ which satisfies*

Ind: $(u, \{a, b\}) \in I \wedge\ a \neq b \Rightarrow (u, a) \asymp_{\mathcal{L}} (u.b, a)$ *[Independence]*

Cfl: $(u, a) \in \mathrm{Pr}(\mathcal{L}) \wedge (u', a) \in \mathrm{Pr}(\mathcal{L}) \wedge u \sim u' \Rightarrow (u, a) \asymp_{\mathcal{L}} (u', a)$ *[Confluence]*

Lab: $(u, a) \asymp_{\mathcal{L}} (v, b) \Rightarrow a = b$ *[Labeling]*

Occ: $u.a \leq v.a \wedge\ (u, a) \asymp_{\mathcal{L}} (v, a) \Rightarrow u = v$ *[Occurrence Separation]*

Ind and Cfl specify which prime intervals should definitely be identified whereas Lab and Occ limit rationally the allowed identifications: Lab ensures that equivalent prime intervals correspond to the same occurrence of action and by Occ no execution sequence allows more than one occurrence of the same event.

Given an equivalence of prime intervals of a local trace language, we can now associate with it a local event structure.

DEFINITION 1.6. *Let $\mathcal{L} = (\Sigma, I, L)$ and let $\asymp_{\mathcal{L}}$ be an equivalence of prime intervals of $\mathcal{L}$. For any word $u \in L$, the set of events in u is $Eve_{\asymp_{\mathcal{L}}}(u) = \{\langle v, b\rangle_{\mathcal{L}} \mid v.b \leq u\}$, where $\langle v, b\rangle_{\mathcal{L}}$ denotes the $\asymp_{\mathcal{L}}$-class of (v, b). The local event structure $\mathfrak{les}_{\asymp_{\mathcal{L}}}(\mathcal{L})$ is the triple $(E, C, \vdash)$ where $C = \{Eve_{\asymp_{\mathcal{L}}}(u) \mid u \in L\}$, $E = \cup C$, and*

$$c \vdash \{e_1, ..., e_n\} \Leftrightarrow \begin{cases} \exists u \in \Sigma^*, \exists a_1, ..., a_n \in \Sigma, (u, \{a_1, ..., a_n\}) \in I \\ \wedge\ Eve_{\asymp_{\mathcal{L}}}(u) = c \wedge\ \forall i \in [1, n], e_i = \langle u, a_i\rangle_{\mathcal{L}} \end{cases}$$

This definition is essentially the same as Definition 2.3 of [4], the only difference being the representation of the independence relation through a local trace language. The proof that $\mathfrak{les}_{\asymp_{\mathcal{L}}}(\mathcal{L})$ is indeed a local event structure is completely analogous to the proof in [4].

2 Generic Categorical Connection

In this section the generic approach towards relating local event structures and local trace languages is outlined. By equipping both classes of objects with suitable behavior-preserving morphisms we obtain the categories $\mathbb{LES}$ and $\mathbb{LTL}$.

Categories $\mathbb{LES}$ and $\mathbb{LTL}$. First, we provide local trace languages with morphisms which preserve sequential executions and independencies.

DEFINITION 2.1. *A (local trace) morphism λ from $\mathcal{L} = (\Sigma, I, L)$ to $\mathcal{L}' = (\Sigma', I', L')$ is a partial function $\lambda : \Sigma \rightharpoonup \Sigma'$ such that*

– $\forall (u,p) \in I,\ (\lambda(u),\lambda(p)) \in I'$;
– $\forall (u,\{a,b\}) \in I: a \neq b \wedge\ \lambda(a)$ *and* $\lambda(b)$ *are both defined* $\Rightarrow \lambda(a) \neq \lambda(b)$.
We note $\mathbb{LTL}$ *the category of LTL provided with these morphisms.*

Note that, due to Axiom LTL_5 of Def. 1.2, $\lambda(L) \subseteq L'$; moreover if u_1 and u_2 are trace equivalent according to I then $\lambda(u_1)$ and $\lambda(u_2)$ are trace equivalent according to I'. Note finally that if two distinct actions a and b are independent after u and if $\lambda(a)$ and $\lambda(b)$ are both defined then they should be independent after $\lambda(u)$ in order to respect concurrency: therefore in this case we require that $\lambda(a) \neq \lambda(b)$.

Next we consider morphisms for local event structures.

DEFINITION 2.2. *A LES morphism* η *from* $\mathcal{E} = (E, C, \vdash)$ *to* $\mathcal{E}' = (E', C', \vdash')$ *is a partial function* $\eta : E \rightharpoonup E'$ *such that* $\forall c \in C,\ \forall p \in \wp_f(E)$: $c \vdash p \Rightarrow \eta(c) \vdash' \eta(p)$.
We note $\mathbb{LES}$ *the category of LES provided with these morphisms.*

Morphisms of local event structures preserve the enabling relation and hence the concurrency between events.

Impossibility Result. In $\mathfrak{ltl}$ we already have a map from the objects of $\mathbb{LES}$ to those of $\mathbb{LTL}$. This map can be extended to a functor from $\mathbb{LES}$ to $\mathbb{LTL}$, because each local event structure morphism $\eta : E \rightharpoonup E'$ from $\mathcal{E}$ to $\mathcal{E}'$ induces a local trace morphism $\mathfrak{ltl}(\eta) : E \rightharpoonup E'$ from $\mathfrak{ltl}(\mathcal{E})$ to $\mathfrak{ltl}(\mathcal{E}')$ defined by $\mathfrak{ltl}(\eta) = \eta$.

LEMMA 2.3. $\mathfrak{ltl}$ *is a functor from* $\mathbb{LES}$ *to* $\mathbb{LTL}$.

This is the expected intuitive translation which extends very simply the classical functor from prime event structures to trace languages. On the other hand, there are in general various ways to map a local trace language to a local event structure, depending on the chosen equivalence of prime intervals. Given a family of such equivalences (one for each local trace language) one could attempt to lift the corresponding family of mappings to a functor from $\mathbb{LTL}$ to $\mathbb{LES}$. As we will see in the examples to follow, for certain families of equivalences of prime intervals this can indeed be done. However, none of these functors can act as a right-adjoint to $\mathfrak{ltl}$ in an adjunction between $\mathbb{LES}$ and $\mathbb{LTL}$. In fact we even have

THEOREM 2.4. *There is no adjunction between* $\mathbb{LES}$ *and* $\mathbb{LTL}$ *with* $\mathfrak{ltl}$ *as the left-adjoint.*

This result directly follows from a result (Th. 3.2) given in Section 3.

Punctuation. Given the impossibility of an adjunction between $\mathbb{LES}$ and $\mathbb{LTL}$ with $\mathfrak{ltl}$ as the left-adjoint, we now investigate the relationships between subcategories. We are particularly interested in having $\mathfrak{ltl}$ as the left-adjoint and a functor based on prime interval respecting maps as the right-adjoint. Using a generic approach which abstracts from concrete equivalences of prime intervals, it is possible to identify conditions on objects and arrows in the categories which lead to subcategories between which coreflections of the desired nature do exist.

Equivalences of prime intervals are essential to separate occurrences of the same action which are perceived as different events (see Axiom Occ of Def. 1.5).

Thus the choice of such an equivalence is a *semantic issue*. In analogy with the use of punctuation in a text to influence its meaning, we call the choice of an equivalence $\asymp_{\mathcal{L}}$ for each local trace language $\mathcal{L}$ a *punctuation*. A natural condition on a punctuation is that it should respect isomorphisms between local trace languages, as otherwise behaviorally equivalent local trace languages could be given a different semantics.

DEFINITION 2.5. *A punctuation is a family of equivalences* $\pi = (\asymp_{\mathcal{L}})_{\mathcal{L}\in\mathbb{LTL}}$ *such that each* $\asymp_{\mathcal{L}}$ *is an equivalence of prime intervals of* $\mathcal{L}$ *and for any isomorphism* $\lambda : \mathcal{L} \to \mathcal{L}'$ *of* $\mathbb{LTL}$*:* $(u,a) \asymp_{\mathcal{L}} (v,b) \Rightarrow (\lambda(u),\lambda(a)) \asymp_{\mathcal{L}'} (\lambda(v),\lambda(b))$.

Thus a punctuation π determines a representation of each local trace language $\mathcal{L}$ as a local event structure $\mathfrak{les}_{\asymp_{\mathcal{L}}}(\mathcal{L})$; hence it provides a translation $\mathfrak{les}_{\pi}$ from local trace languages to local event structures for which $\mathcal{L}$ maps to $\mathfrak{les}_{\asymp_{\mathcal{L}}}(\mathcal{L})$.

Before extending this map to a functor, we restrict the arrows of $\mathbb{LTL}$ to those local trace morphisms that respect the choice of events prescribed by π.

DEFINITION 2.6. *Let* $\pi = (\asymp_{\mathcal{L}})_{\mathcal{L}\in\mathbb{LTL}}$ *be a punctuation. A morphism* $\lambda : \mathcal{L} \to \mathcal{L}'$ *is* π-stable *if* $(u,a) \asymp_{\mathcal{L}} (v,a) \wedge \lambda(a)$ *is defined* $\Rightarrow (\lambda(u),\lambda(a)) \asymp_{\mathcal{L}'} (\lambda(v),\lambda(a))$.

Note that by Def. 2.5, local trace isomorphisms are always stable. Each π-stable local trace morphism λ from $\mathcal{L}$ to $\mathcal{L}'$ induces a local event structure morphism $\mathfrak{les}_{\pi}(\lambda)$ from $\mathfrak{les}_{\pi}(\mathcal{L})$ to $\mathfrak{les}_{\pi}(\mathcal{L}')$ defined by $\mathfrak{les}_{\pi}(\lambda)(\langle u,a\rangle_{\mathcal{L}}) = \langle\lambda(u),\lambda(a)\rangle_{\mathcal{L}'}$. Thus $\mathfrak{les}_{\pi}(\lambda)$ can be extended to a functor between the subcategory of $\mathbb{LTL}$ with only π-stable morphisms and the category $\mathbb{LES}$.

LEMMA 2.7. *Let* $\pi = (\asymp_{\mathcal{L}})_{\mathcal{L}\in\mathbb{LTL}}$ *be a punctuation;* $\mathfrak{les}_{\pi}$ *is a functor from the category of local trace languages with* π*-stable morphisms to* $\mathbb{LES}$.

Thus, with this restriction to π-stable morphisms, we get a morphism preserving way back from local trace languages to local event structures. As the following examples will demonstrate, for some particular punctuations π, this restriction is not a real restriction as all local trace morphisms are π-stable.

Examples. The simplest punctuation is called *Projectivity*; it consists of the equivalences $\pi^p = (\asymp^p_{\mathcal{L}})_{\mathcal{L}\in\mathbb{LTL}}$ such that each $\asymp^p_{\mathcal{L}}$ is the least equivalence over $\mathrm{Pr}(\mathcal{L})$ satisfying the conditions Ind and Cfl of Definition 1.5. Note that each $\asymp^p_{\mathcal{L}}$ also satisfies Lab and Occ so it is the least equivalence of prime intervals of $\mathcal{L}$. This rule of identification was used in [8] to connect prime event structures and Mazurkiewicz traces, and initially in [6] for the connection with 1-safe Petri nets.

The punctuation *History* π^h corresponds to the rule of identification introduced in [3] in order to connect Petri nets with local event structures. In [4] it has been formally defined by $\pi^h = (\asymp^h_{\mathcal{L}})_{\mathcal{L}\in\mathbb{LTL}}$ such that each $\asymp^h_{\mathcal{L}}$ is the least equivalence over $\mathrm{Pr}(\mathcal{L})$ satisfying Ind and Cjc (Conjunction) defined by:
Cjc : $(u,a) \in \mathrm{Pr}(\mathcal{L}) \wedge (u',a) \in \mathrm{Pr}(\mathcal{L}) \wedge \mathrm{Eve}_{\asymp_{\mathcal{L}}}(u) = \mathrm{Eve}_{\asymp_{\mathcal{L}}}(u') \Rightarrow (u,a) \asymp_{\mathcal{L}} (u',a)$
Since every $\asymp^h_{\mathcal{L}}$ is an equivalence of prime intervals, it follows that $\asymp^p_{\mathcal{L}} \subseteq \asymp^h_{\mathcal{L}}$ for all local trace languages $\mathcal{L}$. In general this inclusion is strict: History leads to more identifications of prime intervals than Projectivity (for an example see [4]).

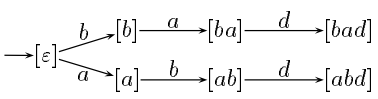

FIG. 1. $\mathcal{L}_1$

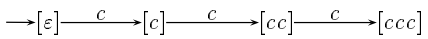

FIG. 2. $\mathcal{L}_2$

The punctuation $\pi^q = (\asymp^q_{\mathcal{L}})_{\mathcal{L}\in\mathbb{LTL}}$ consists of the equivalences $\asymp^q_{\mathcal{L}}$ such that each $\asymp^q_{\mathcal{L}}$ is the least equivalence over $\mathrm{Pr}(\mathcal{L})$ which satisfies Ind and the following condition: $[(u,a)\in\mathrm{Pr}(\mathcal{L})\wedge\ (u',a)\in\mathrm{Pr}(\mathcal{L})\wedge\ \forall x\in\Sigma, |u|_x=|u'|_x]\Rightarrow(u,a)\asymp_{\mathcal{L}}(u',a)$. Thus in the history preceding an occurrence of action only the number of occurrences of each possible action is relevant. This resembles the situation in Petri nets where firing sequences lead to the same marking whenever in these sequences, for each transition its number of occurrences is the same. It is easy to see that for each local trace language $\asymp^h_{\mathcal{L}}\subseteq\asymp^q_{\mathcal{L}}$. In general, History identifies less prime intervals than π^q does. Consider for example the local trace language $\mathcal{L}_1$ depicted in Fig. 1. Here we have $(ab,d)\not\asymp^h_{\mathcal{L}_1}(ba,d)$ but $(ab,d)\asymp^q_{\mathcal{L}_1}(ba,d)$.

We remark now that π^p, π^h, and π^q are particularly simple punctuations because all morphisms are stable w.r.t. them. However this property does not hold for the *Counting* punctuation $\pi^c=(\asymp^c_{\mathcal{L}})_{\mathcal{L}\in\mathbb{LTL}}$ such that $(u,a)\asymp^c_{\mathcal{L}}(v,b)$ iff $a=b\wedge|u|_a=|v|_b$. Thus in Counting two occurrences of an action a are identified, if a has occurred the same number of times in the histories preceding these occurrences. Not all local trace morphisms are π^c-stable. Consider the local trace languages $\mathcal{L}_1$ and $\mathcal{L}_2$ (depicted in Fig. 2) and the morphism $\lambda:\{a,b,d\}\to\{c\}$. This morphism is not π^c-stable because $(b,a)\asymp^c_{\mathcal{L}_1}(\varepsilon,a)$ but $(c,c)\not\asymp^c_{\mathcal{L}_2}(\varepsilon,c)$.

Main Result. In the rest of this section, we consider a fixed punctuation $\pi=(\asymp_{\mathcal{L}})_{\mathcal{L}\in\mathbb{LTL}}$. We establish that the functors $\mathfrak{ltl}$ and $\mathfrak{les}_\pi$ form a coreflection when restricted to certain subcategories of $\mathbb{LES}$ and $\mathbb{LTL}$, determined by π.

First, one essential property of such a connection is that any local event structure $\mathcal{E}$ should be isomorphic to $\mathfrak{les}_\pi\circ\mathfrak{ltl}(\mathcal{E})$. This makes it necessary to cut down on the objects of $\mathbb{LES}$. We single out those local event structures which have the property that for each event of $\mathcal{E}$ all its occurrences in $\mathfrak{ltl}(\mathcal{E})$ are equivalent. This generalizes a similar restriction adopted in [4].

DEFINITION 2.8. *A local event structure $\mathcal{E}$ is* π-singular *if* $\forall u_1.e,u_2.e\in\mathrm{Paths}(\mathcal{E})$, $(u_1,e)\asymp_{\mathfrak{ltl}(\mathcal{E})}(u_2,e)$. $\mathbb{LES}_\pi$ *is the full subcategory of π-singular LES.*

As the next proposition shows, the π-singular local event structures have the desired property; moreover, as far as *finitely branching* local event structures are concerned, π-singularity is an optimal restriction.

PROPOSITION 2.9. *Let $\mathcal{E}=(E,C,\vdash)$ be a local event structure.*

1. *If $\mathcal{E}$ is π-singular then $\mathcal{E}$ is isomorphic to $\mathfrak{les}_\pi\circ\mathfrak{ltl}(\mathcal{E})$;*
2. *If $\mathcal{E}$ is isomorphic to $\mathfrak{les}_\pi\circ\mathfrak{ltl}(\mathcal{E})$ and if $\forall c\in C$, $\mathrm{Card}\{e\in E\mid c\vdash\{e\}\}$ is finite then $\mathcal{E}$ is π-singular.*

Thus now it is necessary to define the subclass of local trace languages which are mapped by $\mathfrak{les}_\pi$ to π-singular local event structures.

DEFINITION 2.10. *A local trace language $\mathcal{L}$ is* π-adequate *if* $\mathfrak{les}_\pi(\mathcal{L})$ *is* π*-singular.*

Despite these restrictions, it may however still be the case that a π-adequate local trace language $\mathcal{L}$ and its associated local event structure $\mathfrak{les}_\pi(\mathcal{L})$ are not close enough. Motivated by a result from [4], we focus on the local trace languages $\mathcal{L} = (\Sigma, I, L)$ such that the equivalence $\asymp_{\mathcal{L}}$ satisfies Cjc and Sym (Symmetry):
Cjc : $(u,a) \in \mathrm{Pr}(\mathcal{L}) \wedge (u',a) \in \mathrm{Pr}(\mathcal{L}) \wedge \mathrm{Eve}_{\asymp_{\mathcal{L}}}(u) = \mathrm{Eve}_{\asymp_{\mathcal{L}}}(u') \Rightarrow (u,a) \asymp_{\mathcal{L}} (u',a)$
Sym : $(u,p) \in I \wedge\ u' \in L \wedge\ \mathrm{Eve}_{\asymp_{\mathcal{L}}}(u) = \mathrm{Eve}_{\asymp_{\mathcal{L}}}(u') \Rightarrow (u',p) \in I$
By Theorem 2.9 of [4], this guarantees that the behaviors described by $\mathcal{L}$ and $\mathfrak{les}_\pi(\mathcal{L})$ are bisimilar. Here this provides sufficient conditions for a coreflection.

DEFINITION 2.11. $\mathbb{LTL}_\pi$ *is the subcategory of* $\mathbb{LTL}$ *whose arrows are the* π*-stable morphisms and whose objects are the* π*-adequate local trace languages that satisfy* Cjc *and* Sym *w.r.t. the punctuation* π*.*

THEOREM 2.12. *The functor* $\mathfrak{les}_\pi : \mathbb{LTL}_\pi \to \mathbb{LES}_\pi$ *is a right-adjoint of* $\mathfrak{tl} : \mathbb{LES}_\pi \to \mathbb{LTL}_\pi$ *which forms a coreflection.*

As we shall argue in the next section this main result of the paper cannot be improved and may be considered to be optimal.

3 Discussion

Singularity and Adequacy. As discussed in the previous section, in order to obtain a coreflection the restriction to π-singular local event structures is, at least for finitely branching local event structures, necessary. Therefore, we had to focus on π-adequate local trace languages. This latter restriction may however be void, as it is for instance the case for History (see [4]). But for the punctuation π^q it is real. Consider, e.g., the local trace language $\mathcal{L}_1$ of Fig. 1. It satisfies Cjc and Sym w.r.t. π^q, but $\mathcal{L}_1$ is not π^q-adequate, since $\mathfrak{les}_{\pi^q}(\mathcal{L}_1)$ is not π^q-singular.

Conjunction and Symmetry. In Definition 2.11 and Theorem 2.12, we chose to focus on local trace languages which satisfy Cjc and Sym w.r.t the punctuation π. As the next proposition shows, this restriction is optimal, at least for *finitely branching* local trace languages.

PROPOSITION 3.1. *Let* $\mathbb{LTL}'$ *be a subcategory of* $\mathbb{LTL}$ *whose arrows are the* π*-stable morphisms between its objects and let* $\mathcal{L} = (\Sigma, I, L)$ *be a local trace language of* $\mathbb{LTL}'$ *such that* $\forall u \in L$, $\mathrm{Card}\{a \in \Sigma \mid u.a \in L\}$ *is finite. If* $\mathfrak{tl} : \mathbb{LES}_\pi \to \mathbb{LTL}'$ *admits* $\mathfrak{les}_\pi$ *as a right-adjoint then* $\mathcal{L}$ *satisfies* Cjc *and* Sym *w.r.t the punctuation* π*.*

This result is similar to the second part of Prop. 2.9 in that it assumes that only a finite number of actions are enabled at any stage of an execution. We should note here that for particular punctuations such as History these assumptions can be omitted in Prop. 2.9 and Prop. 3.1. Thus $\mathbb{LES}_{\pi^h}$ and $\mathbb{LTL}_{\pi^h}$ are the largest full subcategories of $\mathbb{LES}$ and $\mathbb{LTL}$ respectively for which $\mathfrak{tl}$ admits $\mathfrak{les}_{\pi^h}$ as a right-adjoint which forms a coreflection.

Universality. However, the technique of identifying prime intervals to extract events from local event structures constitutes a very general method. In fact no other functors than those based on equivalences of prime intervals could have been used to obtain a coreflection, provided that the local event structures considered include the prime event structures. The following theorem demonstrates that this technique is a *universal strategy* to get all the possible right-adjoints of the natural translation $\mathfrak{ttl}$.

THEOREM 3.2. *Any adjunction between some full subcategories of* $\mathbb{LES}$ *and* $\mathbb{LTL}$ *with* $\mathfrak{ttl}$ *as left-adjoint may be obtained by Theorem 2.12 with an appropriate punctuation — as soon as the local event structures considered include the prime event structures.*

Proof. We consider a full subcategory $\mathbb{LES}'$ of $\mathbb{LES}$ and a full subcategory $\mathbb{LTL}'$ of $\mathbb{LTL}$ such that $\mathfrak{ttl} : \mathbb{LES}' \to \mathbb{LTL}'$ admits a right-adjoint $\mathfrak{les}' : \mathbb{LTL}' \to \mathbb{LES}'$. We assume moreover that $\mathbb{LES}'$ includes the prime event structures. Essentially we exhibit a punctuation π such that $\mathbb{LES}' \subseteq \mathbb{LES}_\pi$, $\mathbb{LTL}' \subseteq \mathbb{LTL}_\pi$, and for any $\mathcal{L} \in \mathbb{LTL}'$, $\mathfrak{les}_\pi(\mathcal{L})$ and $\mathfrak{les}'(\mathcal{L})$ are isomorphic. Hence $\mathfrak{les}'$ and $\mathfrak{les}_\pi$ are equivalent. □

We should note here that this result is the basis of a proof of Th. 2.4 as follows. If $\mathfrak{ttl} : \mathbb{LES} \to \mathbb{LTL}$ admitted a right-adjoint then there would be a punctuation $\pi = (\asymp_\mathcal{L})_{\mathcal{L} \in \mathbb{LTL}}$ such that $\mathbb{LES} = \mathbb{LES}_\pi$ and $\mathbb{LTL} = \mathbb{LTL}_\pi$; furthermore, in that case we easily prove that π must be the Counting punctuation π^c. Yet, we have already remarked that $\mathbb{LTL} \neq \mathbb{LTL}_{\pi^c}$ because some local trace morphisms are not π^c-stable.

References

1. Degano, P., Gorrieri, R., Vigna, S.: *On Relating some Models for Concurrency.* TAPSOFT, LNCS **668** (1993) 15–30
2. Hoogers, P.W., Kleijn, H.C.M., Thiagarajan, P.S.: *A Trace Semantics for Petri Nets.* Information and Computation **117** (1995) 98–114
3. Hoogers, P.W., Kleijn, H.C.M., Thiagarajan, P.S.: *An Event Structure Semantics for General Petri Nets.* Theoretical Computer Science **153** (1996) 129–170
4. Kleijn, H.C.M., Morin, R., Rozoy, B.: *Event Structures for Local Traces.* Electronic Notes in Theoretical Computer Science **16-2** (1998) – 16 pages
5. Mazurkiewicz, A.: *Trace Theory.* Advanced Course on Petri Nets, Bad Honnef, Germany, LNCS **254** (1987) 269–324
6. Nielsen, M., Plotkin, G., Winskel, G.: *Petri nets, events structures and domains, Part I.* Theoretical Computer Science **13** (1981) 85–108
7. Nielsen, M., Sassone, V., Winskel, G.: *Relationships between Models of Concurrency.* LNCS **803** (1994) 425–475
8. Rozoy, B., Thiagarajan, P.S.: *Trace monoids and event structures.* Theoretical Computer Science **91** (1991) 285–313
9. Pierce, B.C.: *Category Theory for Computer Scientists.* (The MIT Press, 1991)
10. Winskel, G., Nielsen, M.: *Models for Concurrency.* In *Handbook of Logic in Computer Science, Vol. 4: Semantic Modelling*, S. Abramsky, D.M. Gabbay and T.S.E. Maibaum, eds (Oxford University Press, Oxford, 1995) 1–148