Understanding Safety Constraints Coalgebraically

Grygoriy Zholtkevych[0000−0002−7515−2143] and Maksym Labzhaniia[0000−0002−2666−3959]

Department of Theoretical and Applied Computer Science
School of Mathematics and Computer Science
V.N. Karazin Kharkiv National University
4, Svobody Sqr., Kharkiv, 61022, Ukraine
g.zholtkevych@karazin.ua; m.labzhaniia@gmail.com

Abstract. Safety constraints are crucial to the development of mission-critical systems. The practice of developing software for systems of this type requires reliable methods for identifying and analysing project artefacts. This paper proposes a coalgebraic approach to understanding behavioural constraints for systems of a kind. The advantage of the proposed approach is that it gives a framework for providing abstract semantic models of the domain-specific languages designed for specifying behavioural constraints.

Keywords: behavioural constraint, safety constraint, coalgebra, final coalgebra, coalgebraical semantic model, final semantics

Acknowledgement. G. Zholtkevych thanks professors R. de Simone, F. Mallet, and L. Liquori for detailed discussions of the problems related to this paper during his internship at Inria Sophia Antipolis - Médi and Campus France for funding this internship.

1 Introduction

A lot of modern technical systems are compound and smart. Moreover, they are hybrid in the sense that ones consisted of both physical and cybernetic (software) components. In other words, we can state that modern technical systems of such a kind are cyber-physical systems (see, for example, [11]). This requires the corresponding approaches to designing these systems. First of all, we need to remark that software complex of such a system contains necessarily reactive components i.e. programs intending rather for providing the required behaviour of the system than for handling data [12]. This is because of the incorrect behaviour of a complex technical system can have serious and some times catastrophic consequences for the system surroundings. Thus, we should classify such systems as safety-critical [4]. As well-known, specification and analysis of the behavioural requirements is the most critical phase under safety-critical systems development process [7] taking into account the fact that a most of system faults and errors are consequences of incorrect specifications or of incomplete analysis. Hence,

we need a dependable and mathematically grounded toolkit for specifying and analysing behavioural constraints for designing such systems.

This paper introduces some general framework for constructing rigorous models of safety constraints, which can be used for constructing domain-specific languages for specifying safety constraints. This framework can be included as a component of the mentioned above development toolkit.

In the paper, we use coalgebraic techniques as the main research methodology. This choice is motivated by the fact that universal coalgebras give an adequate mathematical tool for modelling behaviour of systems [13,9].

Sec. 2 introduces the needed notions and notation for sequences of system notifications.

For completeness and reader convenience, we have collected the used coalgebraic concepts in Sec. 3.

Sec. 4 is devoted to studying concrete endofunctors and properties of the coalgebras corresponding to these endofunctors related to safety constraints.

Finally, Sec. 5 is the central section of the paper. It contains definitions of the target constructions and results providing achieving of the claimed goals.

2 Basic Concepts and Notation

In this section, we assume that X is a finite set with at least two elements. Elements of this set are usually interpreted as system notifications.

A mapping (generally speaking partial) $\boldsymbol{s} : \mathbb{N} \to X$ is called an X-***sequence*** if for any $k \in \mathbb{N}$, $\boldsymbol{s}\,k'$ is defined whenever $\boldsymbol{s}\,k$ is defined and $0 \leq k' < k$.

An X-sequence $\boldsymbol{s}$ is called an X-***word*** if there exists $k \in \mathbb{N}$ such that $\boldsymbol{s}\,k$ is undefined; in contrast, an X-sequence $\boldsymbol{s}$ is called a X-***stream*** if $\boldsymbol{s}\,k$ is defined for all $k \in \mathbb{N}$.

We use the notation X^* for referring to the set of all X-words, and $X^{\mathbb{N}}$ for referring to the set of all X-streams. The set X^* contains the sequence defined nowhere, which is below denoted by $\boldsymbol{\epsilon}$.

We use also the notation X^∞ for referring to the set $X^* \bigcup X^{\mathbb{N}}$, and X^+ for referring to the set of all X-words without the word defined nowhere.

For an X-word $\boldsymbol{u}$, we denote by $|\boldsymbol{u}|$ the minimal natural number such that $\boldsymbol{u}\,|\boldsymbol{u}|$ is undefined.
The value $|\boldsymbol{u}|$ where $\boldsymbol{u} \in X^*$ is called ***length*** of $\boldsymbol{u}$.

As usually, we identify $n \in X$ with the X-word $\boldsymbol{u} \in X^*$ such that $\boldsymbol{u}\,0 = n$ and $\boldsymbol{u}\,k$ is undefined if $k > 0$.

For $\boldsymbol{u} \in X^*$ and $\boldsymbol{s} \in X^\infty$, we denote by $\boldsymbol{us}$ the next X-sequence

$$\boldsymbol{us} = \lambda\, k \in \mathbb{N}\,.\, \begin{cases} \boldsymbol{u}\,k & \text{if } k < |\boldsymbol{u}| \\ \boldsymbol{s}(k - |\boldsymbol{u}|) & \text{otherwise} \end{cases}$$

Below we need in the following set

$$n^{-1} \cdot A = \{\boldsymbol{u} \in X^* \mid n\boldsymbol{u} \in A\} \qquad \text{where } A \subset X^* \text{ and } n \in X.$$

For $\boldsymbol{s} \in X^\infty$ and $m \in \mathbb{N}$, we denote by $\boldsymbol{s}_{m..}$ the next X-sequence

$$\boldsymbol{s}_{m..} = \lambda\, k \in \mathbb{N}\,.\, \begin{cases} \boldsymbol{s}(k+m) & \text{if this value is defined} \\ \text{is undefined} & \text{otherwise} \end{cases}$$

Also for $\boldsymbol{s} \in X^\infty$ and $m, l \in \mathbb{N}$, we denote by $\boldsymbol{s}_{m..l}$ the next X-word

$$\boldsymbol{s}_{m..l} = \lambda\, k \in \mathbb{N}\,.\, \begin{cases} \boldsymbol{s}(k+m) & \text{if this value is defined and } k < l - m \\ \text{is undefined} & \text{otherwise} \end{cases}$$

The principal concepts for our studying is given by the following definitions

Definition 1. *A subset $P \subset X^*$ is called **prefix-free** if $\boldsymbol{u} \in P$ ensures $\boldsymbol{u}_{0..m} \notin P$ whenever $0 \leq m < |\boldsymbol{u}|$.*

Remark 1. If a prefix-free subset of X^* contains $\boldsymbol{\epsilon}$ then this subset is $\{\boldsymbol{\epsilon}\}$. Indeed, if a prefix-free subset of X^* contains both $\boldsymbol{\epsilon}$ and another X-word $\boldsymbol{u}$ then $\boldsymbol{u}_{0..0} = \boldsymbol{\epsilon}$ cannot belong to this subset. This contradiction grounds the remark.

Definition 2 (see [2]). *A **safety constraint** is a subset $S \subset X^{\mathbb{N}}$ such that $\boldsymbol{s} \in S$ if for any $m \in \mathbb{N}$, $\boldsymbol{s}_{0..m} = \boldsymbol{s}'_{0..m}$ for some $\boldsymbol{s}' \in S$.*

3 Coalgebras Preliminaries

In this section, we remind the basic definitions and facts related to the concept of a coalgebra in an arbitrary category. In addition, we give some specific concepts in the case when $\mathbb{C} = \mathbf{Set}$.

Thus, we assume the category $\mathbb{C}$ and the endofunctor $\boldsymbol{F}$ of $\mathbb{C}$ are given and held fixed in this section (general information on category theory can be found in [10,3]).

Definition 3. *A morphism a of $\mathbb{C}$ is called an **$\boldsymbol{F}$-coalgebra** if the equality $\operatorname{cod} a = \boldsymbol{F}(\operatorname{dom} a)$ is fulfilled. In the case, $\operatorname{dom} a$ is called the **carrier** of a and denoted below by $\underline{a}$.*

Definition 4. *Let a and b be $\boldsymbol{F}$-coalgebras then a morphism $f\colon \underline{a} \to \underline{b}$ is called an **$\boldsymbol{F}$-morphism** from a into b (symbolically, $f : a \to b$) if the diagram*

$$\begin{array}{ccc} \underline{a} & \xrightarrow{\;f\;} & \underline{b} \\ {\scriptstyle a}\downarrow & & \downarrow{\scriptstyle b} \\ \boldsymbol{F}\,\underline{a} & \xrightarrow{\;\boldsymbol{F}\,f\;} & \boldsymbol{F}\,\underline{b} \end{array} \qquad \textit{commutes}$$

or, equivalently, the equation $(\boldsymbol{F}\,f)\,a = b\,f$ holds.

Proposition 1. *The class of $\boldsymbol{F}$-coalgebras equipped with $\boldsymbol{F}$-morphisms is a category denoted usually by $\mathbf{Coalg}_{\boldsymbol{F}}(\mathbb{C})$ or $\mathbf{Coalg}_{\boldsymbol{F}}$ if the category $\mathbb{C}$ is clear from the context.*

Final objects of this category are very important for constructing semantic models of programming and specification languages.

Definition 5.
(1) *A terminal object of* $\mathbf{Coalg}_{\boldsymbol{F}}$ *if it exists is called a* ***final*** $\boldsymbol{F}$*-coalgebra, which is denoted by* $\boldsymbol{\nu F}$.
(2) *For any* $\boldsymbol{F}$*-coalgebra* a*, the unique* $\boldsymbol{F}$*-morphism from* a *into* $\boldsymbol{\nu F}$ *is called an* ***anamorphism*** *and denoted by* $[\![a]\!]$.

The concept of bisimulation is a key concept in the theory of universal coalgebras.

Definition 6 (see P. Aczel and N. Mendler [1]). *A bisimulation of* $\boldsymbol{F}$*-coalgebras* a *and* b *is a span* $a \xleftarrow{r_a} r \xrightarrow{r_b} b$ *in category* $\mathbf{Coalg}_{\boldsymbol{F}}(\mathbb{C})$ *such that the span* $\underline{a} \xleftarrow{r_a} \underline{r} \xrightarrow{r_b} \underline{b}$ *in category* $\mathbb{C}$ *is a mono-span i.e. the validity of* $r_a h' = r_a h''$ *and* $r_b h' = r_b h''$ *for any object* c *in* $\mathbb{C}$ *and morphisms* $h', h'' : c \to \underline{r}$ *ensures* $h' = h''$.

The next proposition demonstrates that coalgebra morphisms give the simplest examples of bisimulations.

Proposition 2. *For any* $\boldsymbol{F}$*-coalgebras* a *and* b *and* $\boldsymbol{F}$*-morphism* $f : a \to b$*, the span* $a \xleftarrow{\mathrm{id}_a} a \xrightarrow{f} b$ *is a bisimulation of* a *and* b.

Proof. We need to note only that the span $\underline{a} \xleftarrow{\mathrm{id}_{\underline{a}}} \underline{a} \xrightarrow{f} \underline{b}$ is evidently a mono-span. □

Proposition 3. *Assume that category* $\mathbb{C}$ *is finitely complete and there is a final* $\boldsymbol{F}$*-coalgebra then for any bisimulation* $a \xleftarrow{r_a} r \xrightarrow{r_b} b$ *of* $\boldsymbol{F}$*-coalgebras* a *and* b*, there exists a unique monomorphism* $f : \underline{r} \to P$ *such that* $r_a = p_a f$ *and* $r_b = p_b f$ *where* $\underline{a} \xleftarrow{p_a} P \xrightarrow{p_b} \underline{b}$ *is the pullback of the cospan* $\underline{a} \xrightarrow{[\![a]\!]} \underline{\boldsymbol{\nu F}} \xleftarrow{[\![b]\!]} \underline{b}$.

Proof. Indeed, the definition of an anamorphism ensures the commutativity of the next diagram

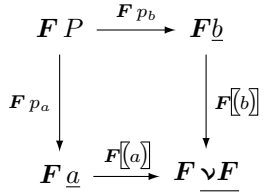
$$\begin{array}{ccc} \underline{r} & \xrightarrow{r_b} & \underline{b} \\ {\scriptstyle r_a}\downarrow & & \downarrow{\scriptstyle [\![b]\!]} \\ \underline{a} & \xrightarrow{[\![a]\!]} & \underline{\boldsymbol{\nu F}} \end{array}$$

i.e. for the pullback $\underline{a} \xleftarrow{p_a} P \xrightarrow{p_b} \underline{b}$ of the cospan $\underline{a} \xrightarrow{[\![a]\!]} \underline{\boldsymbol{\nu F}} \xleftarrow{[\![b]\!]} \underline{b}$, we have the existence of a unique morphism $f : \underline{r} \to P$. One can see that this morphism is a monomorphism taking into account the mono-span and pullback properties. □

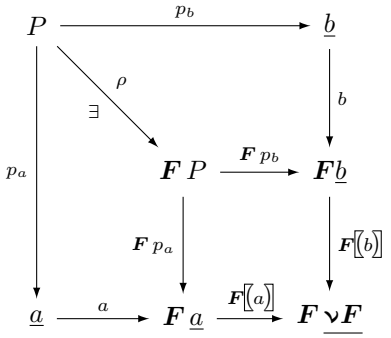
This statement can be inverted for some class of endofunctors.

Proposition 4. *If category $\mathbb{C}$ is finitely complete, endofunctor $\boldsymbol{F}$ preserves pullbacks, and there exists a final $\boldsymbol{F}$-coalgebra then the pullback of the cospan $\underline{a} \xrightarrow{[\![a]\!]} \underline{\boldsymbol{\nu F}} \xleftarrow{[\![b]\!]} \underline{b}$ is the greatest bisimulation of $\boldsymbol{F}$-coalgebras a and b.*

Proof. Indeed, let $\underline{a} \xleftarrow{p_a} P \xrightarrow{p_b} \underline{b}$ be the mentioned above pullback then it is mono-span and the next diagram



is a pullback diagram. Further, note that the outer square of the following diagram commutes.



Indeed,

$$\begin{aligned}
\big(\boldsymbol{F}[\![a]\!]\big)ap_a &= (\boldsymbol{\nu F})[\![a]\!]p_a && \text{considering that } [\![a]\!] \text{ is an } \boldsymbol{F}\text{-morphism} \\
&= (\boldsymbol{\nu F})[\![b]\!]p_b && \text{using the pullback property for } \underline{a} \xleftarrow{p_a} P \xrightarrow{p_b} \underline{b} \\
&= \big(\boldsymbol{F}[\![b]\!]\big)bp_b && \text{considering that } [\![b]\!] \text{ is an } \boldsymbol{F}\text{-morphism.}
\end{aligned}$$

Thus, taking into account the pullback property for the inner subdiagram in the diagram above, one can derive the existence of ρ ensuring the diagram commutativity. But it means that the span $\underline{a} \xleftarrow{p_a} P \xrightarrow{p_b} \underline{b}$ can be lifted up to the corresponding span of coalgebras. □

Another concept used below is the concept of subcoalgebra.

Definition 7. *Let $a \in \mathbf{Coalg}_{\boldsymbol{F}}(\mathbb{C})$ and $j : \underline{j} \to \underline{a}$ be a monomorphism in $\mathbb{C}$ with the domain $\underline{j}$ then an $\boldsymbol{F}$-coalgebra c is called a **subcoalgebra** of a if $\underline{c} = \underline{j}$ and j is lifted upto a coalgebraic monomorphism $j : c \to a$.*

We complete this section by enumerating a series of facts specific for the category **Set** and endofunctors of this category. If $\mathbb{C} = \mathbf{Set}$ then an $\boldsymbol{F}$-coalgebra is called an ***F-system*** due to J. Rutten [13] and the corresponding category is denoted below by $\mathbf{Sys}(\boldsymbol{F})$ instead of $\mathbf{Coalg}_{\boldsymbol{F}}(\mathbf{Set})$.

It is well-known that category **Set** is finitely complete. The important class of endofunctors of the category **Set** is called the class of polynomial endofunctors and defined as follows (see, for example, [9])

(1) a constant endofunctor i.e. an endofunctor $\boldsymbol{C}\,X = \mathsf{C}$ for some set C and $\boldsymbol{C}\,f = \mathrm{id}_{\mathsf{C}}$ is a polynomial endofunctor;
(2) if $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ are polynomial endofunctors then the endofunctor $\boldsymbol{F}\,X = \boldsymbol{F}_1\,X \times \boldsymbol{F}_2\,X$ and $\boldsymbol{F}\,f = \boldsymbol{F}_1\,f \times \boldsymbol{F}_2\,f$ is a polynomial endofunctor;
(3) if $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ are polynomial endofunctors then the endofunctor $\boldsymbol{F}\,X = \boldsymbol{F}_1\,X + \boldsymbol{F}_2\,X$ and $\boldsymbol{F}\,f = \boldsymbol{F}_1\,f + \boldsymbol{F}_2\,f$ is a polynomial endofunctor (here and below, "+" refers to a disjunctive sum of sets).

An important property of polynomial endofunctor is that such an endofunctor preserves pullbacks.

4 Systems as Coalgebras

In this section, we introduce and study some endofunctors, systems related to the endofunctors, and their property. Note that all considered here endofunctors are polynomial.

4.1 Discrete Dynamical Systems

The simplest manner for modelling a discrete dynamical system is to represent it by a pair (X, g) where X is a set called the ***state set*** and $g : X \to X$ is a mapping called the ***dynamics***.

It is evident that g can be considered as $\mathsf{Id}_{\mathbf{Set}}$-coalgebra of the category **Set** and, in this context, X denoted by $\underline{g}$.

Morphisms of such coalgebras are mappings that intertwine the dynamics of the corresponding coalgebras. More precise, for $\mathsf{Id}_{\mathbf{Set}}$-coalgebras g and h, a morphism $f : g \to h$ is a mapping $f : \underline{g} \to \underline{h}$ such that $fg = hf$.

Easy seen that the final coalgebra exists in this case but it is trivial namely $\underline{\nu\mathsf{Id}_{\mathbf{Set}}} = 1$ and $\nu\mathsf{Id}_{\mathbf{Set}} = \mathrm{id}_1$. Thus, taking into account that category **Set** and functor $\mathsf{Id}_{\mathbf{Set}}$ satisfy conditions of Prop. 4 one can conclude that any two dynamical systems are bisimilar.

A less trivial example is given by the endofunctor $\mathsf{T} : \mathbf{Set} \to \mathbf{Set}$ defined as follows (here and below, 1 refers to an one-element set $\{\Downarrow\}$ containing a fault indicator).

$\mathsf{T}\,X = 1 + X$ for any object X of category **Set**;
$\mathsf{T}\,f = \mathrm{id}_1 + f$ for any morphism f of category **Set**.

A system of such a kind is called a ***system with termination***.

The corresponding class of morphisms (see Def. 4) from a T-system g into a T-system h contains a mapping $f : \underline{g} \to \underline{h}$ if and only if for any $x \in \underline{g}$, either $gx = \Downarrow$ and $h(fx) = \Downarrow$ or $gx \neq \Downarrow$ and $f(gx) = h(fx)$.

There is a final $\boldsymbol{\nu}\mathsf{T}$ in the category $\mathbf{Sys}(\mathsf{T})$. This object is structured as follows

$$\underline{\boldsymbol{\nu}\mathsf{T}} = 1 + \mathbb{N} \qquad \text{where } 1 + \mathbb{N} = \mathbb{N} \bigcup \{\infty\}$$

$$\boldsymbol{\nu}\mathsf{T} = \lambda\, x \in 1 + \mathbb{N}\,.\, \begin{cases} \Downarrow & \text{if } x = 0 \\ \infty & \text{if } x = \infty \\ x - 1 & \text{otherwise} \end{cases}$$

For a T-system g, the corresponding anamorphism $[\![g]\!]$ is defined as follows

$$[\![g]\!] = \lambda\, x \in \underline{g}\,.\, \begin{cases} \infty & \text{if } g^{(k)}x \neq \Downarrow \text{ for all } k \in \mathbb{N}_+ \\ \min\{k \in \mathbb{N} \mid g^{(k+1)}x = \Downarrow\} & \text{otherwise} \end{cases}$$

where

$$\begin{aligned} g^{(1)} &= g \\ g^{(k+1)} &= \lambda\, x \in \underline{g}\,.\, \begin{cases} \Downarrow & \text{if } g^{(k)}x = \Downarrow \\ g(g^{(k)}x) & \text{otherwise} \end{cases} \qquad \text{for } k \in \mathbb{N}_+. \end{aligned}$$

Note 1. Everybody familiar with the concept of a monad can easily see that $g^{(k)}$ is the k^{th}-power of g in the corresponding Kleisli category.

4.2 Systems with Output

In this subsection, we consider the class of dynamical systems equipped with a mechanism for the external monitoring of the current system state. We use the term a ***system with output*** for referring to a system of this class. Following the coalgebraic approach, we model systems of this class as coalgebras, which signature is defined by the corresponding endofunctor $\mathsf{S}_{\mathbf{N}} : \mathbf{Set} \to \mathbf{Set}$ that is defined as follows

$\mathsf{S}_{\mathbf{N}}\, X = \mathbf{N} \times X$ for any object X of category $\mathbf{Set}$;
$\mathsf{S}_{\mathbf{N}}\, f = \mathrm{id}_{\mathbf{N}} \times f$ for any objects X and Y of category $\mathbf{Set}$ and a morphism $f : X \to Y$.

In this definition, $\mathbf{N}$ refers to a finite set of possible output values.

Thus, a system with output is a mapping $\sigma : \underline{\sigma} \to \mathsf{S}_{\mathbf{N}}\, \underline{\sigma}$ where $\underline{\sigma}$ is a set called usually the system ***state set***. Taking into account the universality of the product (see, [10, Sec. III.4] or [3, Sec. 2.4]), one can see that an $\mathsf{S}_{\mathbf{N}}$-system σ is uniquely represented as $\sigma = \langle \sigma_{\text{out}}, \sigma_{\text{tr}} \rangle$ where $\sigma_{\text{out}} = \mathrm{pr}_{\mathbf{N}}\, \sigma : \underline{\sigma} \to \mathbf{N}$ and $\sigma_{\text{tr}} = \mathrm{pr}_{\underline{\sigma}}\, \sigma : \underline{\sigma} \to \underline{\sigma}$ called respectively the ***output and transition functions*** of the system.

The general coalgebraic framework (see Def. 4) gives the following concept of an $\mathsf{S}_{\mathbf{N}}$-morphism: for $\mathsf{S}_{\mathbf{N}}$-system σ and τ, a mapping $f : \underline{\sigma} \to \underline{\tau}$ is called an $\mathsf{S}_{\mathbf{N}}$-morphism if $(\tau_{\text{out}})f = \sigma_{\text{out}}$ and $(\tau_{\text{tr}})f = f(\sigma_{\text{tr}})$.

There is a final object $\boldsymbol{\nu}\mathsf{S}_{\mathbf{N}}$ in the category $\mathbf{Sys}(\mathsf{S}_{\mathbf{N}})$. This object is structured as follows

$$\begin{aligned}
\underline{\boldsymbol{\nu}\mathsf{S}_{\mathbf{N}}} &= \mathbf{N}^{\mathbb{N}};\\
(\boldsymbol{\nu}\mathsf{S}_{\mathbf{N}})_{\mathrm{out}} &= \lambda\ s \in \mathbf{N}^{\mathbb{N}}\ .\ s\,0;\\
(\boldsymbol{\nu}\mathsf{S}_{\mathbf{N}})_{\mathrm{tr}} &= \lambda\ s \in \mathbf{N}^{\mathbb{N}}\ .\ \lambda\ k \in \mathbb{N}\ .\ s(k+1).
\end{aligned}$$

For an $\mathbf{N}$-system σ, the corresponding anamorphism $[\![\sigma]\!]$ is defined by the next formula

$$[\![\sigma]\!] = \lambda\ x \in \underline{\sigma}\ .\ \lambda\ k \in \mathbb{N}\ .\ \sigma_{\mathrm{out}}(\sigma_{\mathrm{tr}}^{k}x).$$

Thus, the proposed model allows considering a point of the final $\mathbf{N}$-system carrier as an observed behaviour of the system being studied. It means that we are interested in specifying and analysing constraints that allow distinguishing an admissible and inadmissible system behaviour.

4.3 Detectors of Behavioural Violations

The remaining part of the paper is devoted to demonstrating that the safeness of a subset can be described with using the category-theoretic language.

In this subsection, we introduce some class of systems that used as a tool for distinguishing admissible and inadmissible system behaviours. We refer to systems of this class as detectors of behavioural violations.

This class is determined with the endofunctor $\mathsf{D}_{\mathbf{N}} : \mathbf{Set} \to \mathbf{Set}$ defined as follows

$$\begin{aligned}
&\mathsf{D}_{\mathbf{N}}\,X = (1+X)^{\mathbf{N}} \quad \text{for any object } X \text{ of category } \mathbf{Set};\\
&\mathsf{D}_{\mathbf{N}}\,f = \lambda\ \phi \in (1+X)^{\mathbf{N}}\ .\ \big(\lambda\ n \in \mathbf{N}\ .\ (\mathrm{id}_1 + f)(\phi\,n)\big)\\
&\quad \text{for any objects } X \text{ and } Y \text{ of category } \mathbf{Set} \text{ and a morphism } f : X \to Y.
\end{aligned}$$

We call below a $\mathsf{D}_{\mathbf{N}}$-system a ***detector*** and for a detector $\mathfrak{a}$, we refer to $\underline{\mathfrak{a}}$ as to the detector ***state set***.

A detector morphism f from a detector $\mathfrak{a}$ into a detector $\mathfrak{b}$ is (compare with Def. 4) a mapping $f : \underline{\mathfrak{a}} \to \underline{\mathfrak{b}}$ such that for any $x \in \underline{\mathfrak{a}}$ and $n \in \mathbf{N}$,
(1) $(\mathfrak{a}x)\,n = \Downarrow$ if and only if $(\mathfrak{b}(fx))\,n = \Downarrow$;
(2) if $(\mathfrak{a}x)\,n \neq \Downarrow$ then $\big(\mathfrak{b}(fx)\big)\,n = f\big((\mathfrak{a}x)\,n\big)$.

Proposition 5. *The class* $\mathbf{Sys}(\mathsf{D}_{\mathbf{N}})$ *of* $\mathbf{N}$*-detectors equipped with detector morphisms forms a category.*

Proof is a simple check of the categories axioms. □

The following theorem describes a final detector.

Theorem 1. *There is a final object in category* $\mathbf{Sys}(\mathsf{D}_{\mathbf{N}})$ *that is structured as follows*

$\underline{\boldsymbol{\nu}\mathsf{D}_{\mathbf{N}}}$ *is the set of all prefix-free subsets of* $\mathbf{N}^{+}$*;*

$$\boldsymbol{\nu}\mathsf{D}_{\mathbf{N}} = \lambda\ P \in \underline{\boldsymbol{\nu}\mathsf{D}_{\mathbf{N}}}\ .\ \lambda\ n \in \mathbf{N}\ .\ \begin{cases} \Downarrow & \text{if } n \in P\\ n^{-1}\cdot P & \text{otherwise} \end{cases}.$$

For an a $\mathbf{N}$*-detector, the corresponding anamorphism* $[\![\mathfrak{a}]\!]$ *is defined by the next formula*

$$[\![\mathfrak{a}]\!] = \lambda\, x \in \underline{\mathfrak{a}}\,.\,\{\boldsymbol{u} \in \mathbf{N}^+ \mid \mathfrak{a}^+(x,\boldsymbol{u}) = \Downarrow \text{ and } \mathfrak{a}^+(x,\boldsymbol{u}_{0..k}) \neq \Downarrow \text{ whenever } 0 < k < |\boldsymbol{u}|\}$$

where $\mathfrak{a}^+ : \underline{\mathfrak{a}} \times \mathbf{N}^+ \to 1 + \underline{\mathfrak{a}}$ *defined as follows*

$$\mathfrak{a}^+(x,n) = (\mathfrak{a}x)\,n \qquad x \in \underline{\mathfrak{a}},\ n \in \mathbf{N};$$

$$\mathfrak{a}^+(x,\boldsymbol{u}n) = \begin{cases} \Downarrow & \text{if } \mathfrak{a}^+(x,\boldsymbol{u}) = \Downarrow \\ \big(\mathfrak{a}\mathfrak{a}^+(x,\boldsymbol{u})\big)n & \text{otherwise} \end{cases} \qquad x \in \underline{\mathfrak{a}},\ n \in \mathbf{N},\ \boldsymbol{u} \in \mathbf{N}^+.$$

The theorem can be derived from [8, Lemma 6] but we give a direct proof, which details are used below. The proof is preceded by a statement of the facts being used.

Lemma 1. *For any* $\mathbf{N}$*-detector* $\mathfrak{a}$*,* $x \in \underline{\mathfrak{a}}$*, and* $\boldsymbol{u}, \boldsymbol{v} \in \mathbf{N}^+$*, the equation*

$$\mathfrak{a}^+(x,\boldsymbol{u}\boldsymbol{v}) = \begin{cases} \Downarrow & \text{if } \mathfrak{a}^+(x,\boldsymbol{u}) = \Downarrow \\ \mathfrak{a}^+\big(\mathfrak{a}^+(x,\boldsymbol{u}),\boldsymbol{v}\big) & \text{otherwise} \end{cases} \tag{1}$$

holds.

Proof. If $\mathfrak{a}^+(x,\boldsymbol{u}) = \Downarrow$ then $\mathfrak{a}^+(x,\boldsymbol{u}\boldsymbol{v}) = \Downarrow$ by definition of $\mathfrak{a}^+$. Hence we below assume $\mathfrak{a}^+(x,\boldsymbol{u}) \neq \Downarrow$.
Let us use induction on $\boldsymbol{v}$.
For $\boldsymbol{v} = n \in \mathbf{N}$, (1) follows from the definition of $\mathfrak{a}^+$.
Assume that $\boldsymbol{v} = \boldsymbol{v}'n$, $n \in \mathbf{N}$, and (1) holds for $\boldsymbol{v}'$.
Then under assumption that $\mathfrak{a}^+(x,\boldsymbol{u}\boldsymbol{v}') = \Downarrow$, we have

$$\mathfrak{a}^+(x,\boldsymbol{u}(\boldsymbol{v}'n)) = \mathfrak{a}^+(x,(\boldsymbol{u}\boldsymbol{v}')n) = \Downarrow$$

by definition of $\mathfrak{a}^+$. Hence, $\mathfrak{a}^+(x,\boldsymbol{u}\boldsymbol{v}) = \Downarrow$.
In the other side, we have $\mathfrak{a}^+(\mathfrak{a}^+(x,\boldsymbol{u}),\boldsymbol{v}') = \mathfrak{a}^+(x,\boldsymbol{u}\boldsymbol{v}') = \Downarrow$ by induction hypothesis. But then $\mathfrak{a}^+(\mathfrak{a}^+(x,\boldsymbol{u}),\boldsymbol{v}) = \mathfrak{a}^+(\mathfrak{a}^+(x,\boldsymbol{u}),\boldsymbol{v}'n) = \Downarrow$ by definition of $\mathfrak{a}^+$.
Hence, we have $\mathfrak{a}^+(x,\boldsymbol{u}\boldsymbol{v}) = \mathfrak{a}^+(\mathfrak{a}^+(x,\boldsymbol{u}),\boldsymbol{v})$.
In contrary, assumption that $\mathfrak{a}^+(x,\boldsymbol{u}\boldsymbol{v}') \neq \Downarrow$ gives

$$\begin{aligned} \mathfrak{a}^+(x,\boldsymbol{u}\boldsymbol{v}) &= \mathfrak{a}^+(x,\boldsymbol{u}(\boldsymbol{v}'n)) = \mathfrak{a}^+(x,(\boldsymbol{u}\boldsymbol{v}')n) \\ &= \big(\mathfrak{a}\,\mathfrak{a}^+(x,\boldsymbol{u}\boldsymbol{v}')\big)\,n && \text{by definition of } a^+ \\ &= \big(\mathfrak{a}\,\mathfrak{a}^+(\mathfrak{a}^+(x,\boldsymbol{u}),\boldsymbol{v}')\big)\,n && \text{by induction hypothesis} \\ &= \mathfrak{a}^+(\mathfrak{a}^+(x,\boldsymbol{u}),\boldsymbol{v}'n) = \mathfrak{a}^+(\mathfrak{a}^+(x,\boldsymbol{u}),\boldsymbol{v}) && \text{by definition of } a^+. \end{aligned}$$

Thus, the lemma is proved. □

Lemma 2. *If* $P \subset \mathbf{N}^+$ *is prefix-free then* $n^{-1} \cdot P \subset \mathbf{N}^+$ *and it is prefix-free whenever* $n \in \mathbf{N}$ *and* $n \notin P$*.*

Proof. Firstly, $n \notin P$ ensures $\boldsymbol{\epsilon} \notin n^{-1}\cdot P$ i.e. $n^{-1}\cdot P \subset \mathbf{N}^+$. Further, if $\boldsymbol{u} \in n^{-1}\cdot P$ and $\boldsymbol{u}_{0..m} \in n^{-1}\cdot P$ for $0 \le m < |\boldsymbol{u}|$ then $n\boldsymbol{u} \in P$ and $n\boldsymbol{u}_{0..m} \in P$. Taking into account $n\boldsymbol{u}_{0..m} = (n\boldsymbol{u})_{0..m+1}$, one can obtain a contradiction, which completes the proof. □

Lemma 3. *Any prefix-free subset P of $\mathbf{N}^+$ can be represented as the following disjunctive union*

$$\begin{gathered} P = \mathbf{N}_P + \sum_{n\in\mathbf{N}\setminus\mathbf{N}_P} n\cdot P_n \qquad \textit{where} \\ \mathbf{N}_P = \{n \in \mathbf{N} \mid n \in P\} \textit{ and } P_n = n^{-1}\cdot P. \end{gathered} \tag{2}$$

Proof. Assume $\boldsymbol{u} \in P$ then either $\boldsymbol{u} = n$ for some $n \in \mathbf{N}$ or $|\boldsymbol{u}| > 1$. In the first case, we have $\boldsymbol{u} \in \mathbf{N}_P$ and, therefore, $\boldsymbol{u}$ belongs to the right side of (2). In another case, $\boldsymbol{u} = (\boldsymbol{u}\,0)\boldsymbol{u}_{1..}$ where $(\boldsymbol{u}\,0) \notin \mathbf{N}_P$ and $|\boldsymbol{u}_{1..}| > 0$ i.e. $\boldsymbol{u}_{1..} \in (\boldsymbol{u}\,0)^{-1}\cdot P$ and $\boldsymbol{u}$ belongs to the right side of (2).
Now assume $\boldsymbol{u}$ belongs to the right side of (2) then either $\boldsymbol{u} \in \mathbf{N}_P$ or $\boldsymbol{u} \in \sum\limits_{n\in\mathbf{N}\setminus\mathbf{N}_P} n\cdot P_n$. In the first case, we have $\boldsymbol{u} = n \in \mathbf{N}_P$ i.e. $\boldsymbol{u} \in P$ by definition of $\mathbf{N}_P$. In another case, $\boldsymbol{u} \in n\cdot P_n$ where $n \in \mathbf{N}\setminus\mathbf{N}_P$. Hence, $\boldsymbol{u} = n\boldsymbol{u}_{1..}$ and $\boldsymbol{u}_{1..} \in n^{-1}\cdot P$. The last means that $\boldsymbol{u} \in P$. □

Now all is ready for proving Theorem 1.

Proof (Proof of Theorem 1). Firstly, $\boldsymbol{\nu}\mathsf{D}_\mathbf{N} : \underline{\boldsymbol{\nu}\mathsf{D}_\mathbf{N}} \to \mathsf{D}_\mathbf{N}\,\underline{\boldsymbol{\nu}\mathsf{D}_\mathbf{N}}$ due to Lemma 2. Hence, $\boldsymbol{\nu}\mathsf{D}_\mathbf{N}$ is an $\mathbf{N}$-detector.
Further, let us show that the mapping $[\![\mathfrak{a}]\!] : \underline{\mathfrak{a}} \to \underline{\boldsymbol{\nu}\mathsf{D}_\mathbf{N}}$ defined above for any $\mathbf{N}$-detector $\mathfrak{a}$ is an $\mathbf{N}$-detector morphism.
If $(\mathfrak{a}x)\,n = \Downarrow$ then $n \in [\![\mathfrak{a}]\!]x$ i.e. $\big(\boldsymbol{\nu}\mathsf{D}_\mathbf{N}([\![\mathfrak{a}]\!]x)\big)n = \Downarrow$.
Conversely, if $\big(\boldsymbol{\nu}\mathsf{D}_\mathbf{N}([\![\mathfrak{a}]\!]x)\big)n = \Downarrow$ then $n \in [\![\mathfrak{a}]\!]x$ i.e. $(\mathfrak{a}x)\,n = \Downarrow$.
If $(\mathfrak{a}x)\,n \ne \Downarrow$ then either $[\![\mathfrak{a}]\!]x = \varnothing$ or this equation is wrong.
In the first case, $\mathfrak{a}^+(x,\boldsymbol{u}) \ne \Downarrow$ for any $\boldsymbol{u} \in \mathbf{N}^+$ and, therefore, $[\![\mathfrak{a}]\!]\big((\mathfrak{a}x)\,n\big) = \varnothing$ too. Also, we have

$$\big(\boldsymbol{\nu}\mathsf{D}_\mathbf{N}([\![\mathfrak{a}]\!]x)\big)n = (\boldsymbol{\nu}\mathsf{D}_\mathbf{N}\,\varnothing)n = n^{-1}\cdot\varnothing = \varnothing = [\![\mathfrak{a}]\!]\big((\mathfrak{a}x)\,n\big).$$

In another case, we have both $(\mathfrak{a}x)\,n \ne \Downarrow$ and $[\![\mathfrak{a}]\!]x \ne \varnothing$.
If $\boldsymbol{u} \in \big(\boldsymbol{\nu}\mathsf{D}_\mathbf{N}([\![\mathfrak{a}]\!]x)\big)n = n^{-1}\cdot([\![\mathfrak{a}]\!]x)$ then $n\boldsymbol{u} \in [\![\mathfrak{a}]\!]x$ i.e. $\mathfrak{a}^+(x,n\boldsymbol{u}) = \Downarrow$ but $\mathfrak{a}^+(x,n\boldsymbol{u}_{0..m}) \ne \Downarrow$ whenever $m < |\boldsymbol{u}|$. Hence, for any $m \le |\boldsymbol{u}|$ and $\boldsymbol{v} = \boldsymbol{u}_{0..m}$, we have

$$\mathfrak{a}^+(x,n\boldsymbol{v}) = \mathfrak{a}^+(\mathfrak{a}^+(x,n),\boldsymbol{v}) = \mathfrak{a}^+((\mathfrak{a}x)\,n,\boldsymbol{v}) \tag{3}$$

due to Lemma 1 and by definition of $\mathfrak{a}^+$. Hence, one can conclude that $\boldsymbol{u} \in [\![\mathfrak{a}]\!]\big((\mathfrak{a}x)\,n\big)$.
Conversely, if $\boldsymbol{u} \in [\![\mathfrak{a}]\!]\big((\mathfrak{a}x)\,n\big)$ then (3) guarantees $\mathfrak{a}^+(x,n\boldsymbol{u}) = \Downarrow$ but $\mathfrak{a}^+(x,n\boldsymbol{u}_{0..m}) \ne \Downarrow$ whenever $m < |\boldsymbol{u}|$. It means that $\boldsymbol{u} \in n^{-1}\cdot([\![\mathfrak{a}]\!]x) = \big(\boldsymbol{\nu}\mathsf{D}_\mathbf{N}([\![\mathfrak{a}]\!]x)\big)n$.

Thus, we have checked the $\mathbf{N}$-detector morphism properties for $[\![\mathfrak{a}]\!]$.
For completing the proof, we need to show that $[\![\mathfrak{a}]\!]$ is the only $\mathbf{N}$-detector morphism from $\mathfrak{a}$ into $\nu\mathsf{D}_{\mathbf{N}}$ i.e. $f = [\![\mathfrak{a}]\!]$ for any $f : \mathfrak{a} \to \nu\mathsf{D}_{\mathbf{N}}$.
Assume $fx = \varnothing$ then we have the next chain of equivalent statements

$fx = \varnothing$	
$n^{-1} \cdot (fx) = \varnothing$ for all $n \in \mathbf{N}$	
$\big(\nu\mathsf{D}_{\mathbf{N}}(fx)\big)\, n = \varnothing$ for all $n \in \mathbf{N}$	by definition of $\nu\mathsf{D}_{\mathbf{N}}$
$(\mathfrak{a}x)\, n \neq \Downarrow$ and $f\big((\mathfrak{a}x)\, n\big) = \varnothing$ for any $n \in \mathbf{N}$	due to the $\mathbf{N}$-morphism properties
$\mathfrak{a}^+(x, \boldsymbol{u}) \neq \Downarrow$ for all $\boldsymbol{u} \in \mathbf{N}^+$	by induction on $\boldsymbol{u}$
$[\![\mathfrak{a}]\!]x = \varnothing$	by definition of $[\![\mathfrak{a}]\!]$.

Hence, $fx = \varnothing$ iff $[\![\mathfrak{a}]\!]x = \varnothing$.
Now using induction on $|\boldsymbol{u}|$, prove that $\boldsymbol{u} \in fx$ if and only if $\boldsymbol{u} \in [\![\mathfrak{a}]\!]x$.
If $|\boldsymbol{u}| = 1$ then we have the next chain of equivalent statements.

$n \in fx$ for some $n \in \mathbf{N}$	
$\big(\nu\mathsf{D}_{\mathbf{N}}(fx)\big)n = \Downarrow$	by definition of $\nu\mathsf{D}_{\mathbf{N}}$
$(\mathfrak{a}x)\, n = \Downarrow$	due to the $\mathbf{N}$-morphism properties
$n \in [\![\mathfrak{a}]\!]x$	by definition of $[\![\mathfrak{a}]\!]$.

Hence, $\boldsymbol{u} \in fx$ iff $\boldsymbol{u} \in [\![\mathfrak{a}]\!]x$ for any $\boldsymbol{u} \in \mathbf{N}^+$ such that $|\boldsymbol{u}| = 1$.
If $|\boldsymbol{u}| = m > 1$ and the required statement is true for $\boldsymbol{v} \in \mathbf{N}^+$ such that $|\boldsymbol{v}| < m$ then we have the next chain of equivalent statements.

$\boldsymbol{u} \in fx$	
$\boldsymbol{u}_{1..} \in (\boldsymbol{u}\, 0)^{-1} \cdot (fx)$	
$\boldsymbol{u}_{1..} \in \big(\nu\mathsf{D}_{\mathbf{N}}(fx)\big)(\boldsymbol{u}\, 0)$	by definition of $\nu\mathsf{D}_{\mathbf{N}}$ and due to Lemma 3
$\boldsymbol{u}_{1..} \in f\big((\mathfrak{a}x)\,(\boldsymbol{u}\, 0)\big)$	due to the $\mathbf{N}$-morphism properties
$\boldsymbol{u}_{1..} \in [\![\mathfrak{a}]\!]\big((\mathfrak{a}x)(\boldsymbol{u}\, 0)\big)$	by induction hypothesis
$\mathfrak{a}^+\big((ax)(\boldsymbol{u}\, 0), \boldsymbol{u}_{1..}\big) = \Downarrow$ but	
$\quad \mathfrak{a}^+\big((\mathfrak{a}x)(\boldsymbol{u}\, 0), \boldsymbol{u}_{1..k}\big) \neq \Downarrow$ whenever $k < m$	by definition of $[\![a]\!]$
$\mathfrak{a}^+\big(x, (\boldsymbol{u}\, 0)\boldsymbol{u}_{1..}\big) = \Downarrow$ but	
$\quad \mathfrak{a}^+\big(x, (\boldsymbol{u}\, 0)\boldsymbol{u}_{1..k}\big) \neq \Downarrow$ whenever $k < m$	due to Lemma 1
$\boldsymbol{u} \in [\![\mathfrak{a}]\!]x$	by definition of $[\![\mathfrak{a}]\!]$.

Thus, $f = [\![\mathfrak{a}]\!]$. □

Below we need a characterisation of the subsets of $\underline{\nu\mathsf{D}_{\mathbf{N}}}$ that are carriers of subcoalgebras. The next proposition gives this characterisation.

Proposition 6. *A subset $C \subset \boldsymbol{\nu}\mathsf{D}_{\mathbf{N}}$ is the carrier of a subcoalgebras of the final detector defined by the natural embedding $j_C : C \to \underline{\boldsymbol{\nu}\mathsf{D}_{\mathbf{N}}}$ if and only if the condition*

$$\text{for any } P \in C \text{ and } n \in \mathbf{N}, \quad \text{either} \quad n \in P \quad \text{or} \quad n^{-1} \cdot P \in C$$

is fulfilled.

Proof boils down to simply checking the requirements of definitions. □

5 Coalgebraic Understanding Safety Constraints

The general coalgebraic framework for recognising violations of the behaviour of a system with output is introduced and studied in this section.

5.1 Functor Join and Its Properties

This subsection defines bifunctor Join (see Theorem 2), which is the key tool for our studying. The importance of this bifunctor is related to the fact it preserves bisimulations (see Theorem 3).

Firstly assuming σ and $\mathfrak{a}$ is a system with output $\mathbf{N}$ and an $\mathbf{N}$-detector respectively, one can define the system with termination $\mathsf{Join}(\sigma, \mathfrak{a}) : \underline{\sigma} \times \underline{\mathfrak{a}} \to \mathsf{T}(\underline{\sigma} \times \underline{\mathfrak{a}})$ as follows

$$\mathsf{Join}(\sigma, \mathfrak{a}) = \lambda\ (x, y) \in \underline{\sigma} \times \underline{\mathfrak{a}}\ .\ \begin{cases} \Downarrow & \text{if } (\mathfrak{a}y)(\sigma_{\mathrm{out}}\, x) = \Downarrow \\ \langle\, \sigma_{\mathrm{tr}}\, x, (\mathfrak{a}y)(\sigma_{\mathrm{out}}\, x) \rangle & \text{otherwise} \end{cases} \tag{4a}$$

Further for systems σ and τ with output $\mathbf{N}$ and $\mathbf{N}$-detectors $\mathfrak{a}$ and $\mathfrak{b}$, an $\mathsf{S}_{\mathbf{N}}$-morphism $f : \sigma \to \tau$, and a detector morphism $g : \mathfrak{a} \to \mathfrak{b}$, we define the mapping $\mathsf{Join}(f, g) : \underline{\sigma} \times \underline{\mathfrak{a}} \to \underline{\tau} \times \underline{\mathfrak{b}}$ by the formula

$$\mathsf{Join}(f, g) = f \times g. \tag{4b}$$

Theorem 2. *Rules (4) determine a bifunctor* $\mathsf{Join} : \mathbf{Sys}(\mathsf{S}_{\mathbf{N}}) \times \mathbf{Sys}(\mathsf{D}_{\mathbf{N}}) \to \mathbf{Sys}(\mathsf{T})$.

Proof. Firstly, let us check that for any $f : \sigma \to \tau$ and $g : \mathfrak{a} \to \mathfrak{b}$ where $\sigma, \tau \in \mathbf{Sys}(\mathsf{S}_{\mathbf{N}})$ and $a, b \in \mathbf{Sys}(\mathsf{D}_{\mathbf{N}})$, $\mathsf{Join}(f, g)$ is a T-morphism from $\mathsf{Join}(\sigma, \mathfrak{a})$ into $\mathsf{Join}(\tau, \mathfrak{b})$.
Indeed, let $\big(\mathsf{Join}(\sigma, \mathfrak{a})\big)(x, y) = \Downarrow$ where $x \in \underline{\sigma}$, $y \in \underline{\mathfrak{a}}$ then we have the next chain of equivalent statements

$$\begin{aligned}
&\big(\mathsf{Join}(\sigma, \mathfrak{a})\big)\langle x, y\rangle = \Downarrow && \\
&(\mathfrak{a}y)(\sigma_{\mathrm{out}} x) = \Downarrow && \text{by definition of } \mathsf{Join}(\sigma, \mathfrak{a}) \\
&(\mathfrak{a}y)(\tau_{\mathrm{out}}(fx)) = \Downarrow && \text{due to } f \text{ is an } \mathsf{S}_{\mathbf{N}}\text{-morphism} \\
&\big(\mathfrak{b}(gy)\big)\big(\tau_{\mathrm{out}}(fx)\big) = \Downarrow && \text{due to } g \text{ is a } \mathsf{D}_{\mathbf{N}}\text{-morphism} \\
&\big(\mathsf{Join}(\tau, \mathfrak{b})\big)\langle fx, gy\rangle = \Downarrow && \text{by definition of } \mathsf{Join}(\tau, \mathfrak{b}) \\
&\big(\mathsf{Join}(\tau, \mathfrak{b})\big)\big(\mathsf{Join}(f, g)\langle x, y\rangle\big) = \Downarrow && \text{by definition of } \mathsf{Join}(f, g)
\end{aligned}$$

Now assume that $\big(\mathsf{Join}(\sigma,\mathfrak{a})\big)\langle x,y\rangle \neq \Downarrow$ (it means $(\mathfrak{a}y)(\sigma_{\mathrm{out}}x) \neq \Downarrow$) where $x \in \underline{\sigma}$, $y \in \underline{\mathfrak{a}}$ then

$$\begin{aligned}\big(\mathsf{Join}(f,g)\big)\Big(\big(\mathsf{Join}(\sigma,\mathfrak{a})\big)\langle x,y\rangle\Big) &= \big(\mathsf{Join}(f,g)\big)\big\langle \sigma_{\mathrm{tr}}x, (\mathfrak{a}y)(\sigma_{\mathrm{out}}x)\big\rangle \\ &= \big\langle f(\sigma_{\mathrm{tr}}x), g\big((\mathfrak{a}y)(\sigma_{\mathrm{out}}x)\big)\big\rangle\end{aligned}$$

by definitions of $\mathsf{Join}(\sigma,\mathfrak{a})$ and $\mathsf{Join}(f,g)$.
Further,

$$\begin{aligned}\big(\mathsf{Join}(f,g)\big)\Big(\big(\mathsf{Join}(\sigma,\mathfrak{a})\big)\langle x,y\rangle\Big) &= \big\langle \tau_{\mathrm{tr}}(fx), g\big((\mathfrak{a}y)\big(\tau_{\mathrm{out}}(fx)\big)\big)\big\rangle \\ &= \big\langle \tau_{\mathrm{tr}}(fx), \big(\mathfrak{b}(gy)\big)\big(\tau_{\mathrm{out}}(fx)\big)\big\rangle\end{aligned}$$

taking into account that f is an $\mathsf{S}_{\mathbf{N}}$-morphism, and g is a $\mathsf{D}_{\mathbf{N}}$-morphism.
Finally,

$$\begin{aligned}\big(\mathsf{Join}(f,g)\big)\Big(\big(\mathsf{Join}(\sigma,\mathfrak{a})\big)\langle x,y\rangle\Big) &= \big(\mathsf{Join}(\tau,\mathfrak{b})\big)\big\langle fx, gy\big\rangle \\ &= \big(\mathsf{Join}(\tau,\mathfrak{b})\big)\big(\mathsf{Join}(f,g)\langle x,y\rangle\big)\end{aligned}$$

by definitions of $\mathsf{Join}(\tau,\mathfrak{b})$ and $\mathsf{Join}(f,g)$ respectivelly.
Thus, $\mathsf{Join}(f,g)$ is a T-morphism (see Subsec. 4.1).
Taking into account rule (4b) one can conclude that Join is a bifunctor. □

Theorem 3. *Let $\sigma \xleftarrow{p_\sigma} \rho \xrightarrow{p_\tau} \tau$ be a bisimulation of systems σ and τ with output $\mathbf{N}$ and $\mathfrak{a} \xleftarrow{q_\mathfrak{a}} \mathfrak{r} \xrightarrow{q_\mathfrak{b}} \mathfrak{b}$ be a bisimulation of $\mathbf{N}$-detectors $\mathfrak{a}$ and $\mathfrak{b}$ then $\mathsf{Join}(\rho,\mathfrak{r})$ is a bisimulation of $\mathsf{Join}(\sigma,\mathfrak{a})$ and $\mathsf{Join}(\tau,\mathfrak{b})$.*

Proof. Let us consider the span

$$\mathsf{Join}(\sigma,\mathfrak{a}) \xleftarrow{\mathsf{Join}(p_\sigma,q_\mathfrak{a})} \mathsf{Join}(\rho,\mathfrak{r}) \xrightarrow{\mathsf{Join}(p_\tau,q_\mathfrak{b})} \mathsf{Join}(\tau,\mathfrak{b}).$$

It is sufficient to show that the corresponding span $\underline{\sigma}\times\underline{\mathfrak{a}} \xleftarrow{p_\sigma\times q_\mathfrak{a}} \underline{\rho}\times\underline{\mathfrak{r}} \xrightarrow{p_\tau\times q_\mathfrak{b}} \underline{\tau}\times\underline{\mathfrak{b}}$ in **Set** is a mono-span.
Assume some mappings $g,h : S \to \underline{\rho}\times\underline{\mathfrak{r}}$ with common domain S satisfy the equations $(p_\sigma\times q_\mathfrak{a})g = (p_\sigma\times q_\mathfrak{a})h$ and $(p_\tau\times q_\mathfrak{b})g = (p_\tau\times q_\mathfrak{b})h$. These mappings can be uniquely represented as follows $g = \langle g_{\underline{\rho}}, g_{\underline{\mathfrak{r}}}\rangle$ and $h = \langle h_{\underline{\rho}}, h_{\underline{\mathfrak{r}}}\rangle$ respectively where $g_{\underline{\rho}}, h_{\underline{\rho}} : S \to \underline{\rho}$ and $g_{\underline{\mathfrak{r}}}, h_{\underline{\mathfrak{r}}} : S \to \underline{\mathfrak{r}}$.
Taking into account that

$$\begin{aligned}&(p_\sigma\times q_\mathfrak{a})g = (p_\sigma\times q_\mathfrak{a})\langle g_{\underline{\rho}}, g_{\underline{\mathfrak{r}}}\rangle = \langle p_\sigma g_{\underline{\rho}}, q_\mathfrak{a} g_{\underline{\mathfrak{r}}}\rangle \qquad \text{and} \\ &(p_\sigma\times q_\mathfrak{a})h = (p_\sigma\times q_\mathfrak{a})\langle h_{\underline{\rho}}, h_{\underline{\mathfrak{r}}}\rangle = \langle p_\sigma h_{\underline{\rho}}, q_\mathfrak{a} h_{\underline{\mathfrak{r}}}\rangle\end{aligned}$$

we have $\langle p_\sigma g_{\underline{\rho}}, q_\mathfrak{a} g_{\underline{\mathfrak{r}}}\rangle = \langle p_\sigma h_{\underline{\rho}}, q_\mathfrak{a} h_{\underline{\mathfrak{r}}}\rangle$ i.e. $p_\sigma g_{\underline{\rho}} = p_\sigma h_{\underline{\rho}}$ and $q_\mathfrak{a} g_{\underline{\mathfrak{r}}} = q_\mathfrak{a} h_{\underline{\mathfrak{r}}}$ due to properties of product. Similarly, we have $p_\tau g_{\underline{\rho}} = p_\tau h_{\underline{\rho}}$ and $q_\mathfrak{b} g_{\underline{\mathfrak{r}}} = q_\mathfrak{b} h_{\underline{r}}$ from

condition $(p_\tau \times q_\mathfrak{b})g = (p_\tau \times q_\mathfrak{b})h$.
Then one can derive $g_{\underline{\rho}} = h_{\underline{\rho}}$ using the conditions $p_\sigma g_{\underline{\rho}} = p_\sigma h_{\underline{\rho}}$ and $p_\tau g_{\underline{\rho}} = p_\tau h_{\underline{\rho}}$ and the bisimulation $\sigma \xleftarrow{p_\sigma} \rho \xrightarrow{p_\tau} \tau$.
Similarly, we have $g_{\underline{\mathfrak{r}}} = h_{\underline{\mathfrak{r}}}$ due to the conditions $q_\mathfrak{a} g_{\underline{\mathfrak{r}}} = q_\mathfrak{a} h_{\underline{\mathfrak{r}}}$ and $q_\mathfrak{b} g_{\underline{\mathfrak{r}}} = q_\mathfrak{b} h_{\underline{\mathfrak{r}}}$ and the bisimulation $\mathfrak{a} \xleftarrow{q_\mathfrak{a}} \mathfrak{r} \xrightarrow{q_\mathfrak{b}} \mathfrak{b}$.
Thus, $g = h$ and, therefore, the considering span $\underline{\sigma} \times \underline{\mathfrak{a}} \xleftarrow{p_\sigma \times q_\mathfrak{a}} \underline{\rho} \times \underline{\mathfrak{r}} \xrightarrow{p_\tau \times q_\mathfrak{b}} \underline{\tau} \times \underline{\mathfrak{b}}$ is realy a mono-span. □

5.2 Safety Constraints and Detectors

This subsection is central to the paper. Here we establish an association between $\mathbf{N}$-detectors and some subsets of $\mathbf{N}^\mathbb{N}$. Further, we prove this class of subsets is exactly the class of safety constraints.

First of all for $\boldsymbol{s} \in \mathbf{N}^\mathbb{N}$, let us define the following system $[\boldsymbol{s}]$ with output namely

$$\underline{[\boldsymbol{s}]} = \{\boldsymbol{s}_{k..} \mid k \in \mathbb{N}\} \quad \text{and} \quad [\boldsymbol{s}] = \lambda\, \boldsymbol{t} \in \underline{[\boldsymbol{s}]}\,.\,\langle\, \boldsymbol{t}\, 0, \boldsymbol{t}_{1..}\rangle.$$

Further for any $\mathbf{N}$-detector $\mathfrak{a}$ and $x \in \underline{\mathfrak{a}}$, let us define the following set

$$[\![\mathfrak{a}]\!]_x = \{\boldsymbol{s} \in \mathbf{N}^\mathbb{N} \mid [\![\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})]\!]\langle \boldsymbol{s}, x\rangle = \infty\}.$$

The next simple fact is useful below.

Lemma 4. *For $\boldsymbol{s} \in \mathbf{N}^\mathbb{N}$, an $\mathbf{N}$-detector $\mathfrak{a}$, and $x \in \underline{\mathfrak{a}}$,*

$$\big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)^{(m)}\langle \boldsymbol{s}, x\rangle = \begin{cases} \Downarrow & \text{if } \mathfrak{a}^+(x, \boldsymbol{s}_{0..m}) = \Downarrow \\ \langle \boldsymbol{s}_{m..}, \mathfrak{a}^+(x, \boldsymbol{s}_{0..m})\rangle & \text{otherwise} \end{cases} \qquad \text{for } m > 0.$$

Proof. For proving we apply induction on m. If $m = 1$ then the equation holds due to (4a).
Now assume the equation holds for some $m > 1$.
Let $\big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)^{(m)}\langle \boldsymbol{s}, x\rangle = \Downarrow$ then $\mathfrak{a}^+(x, \boldsymbol{s}_{0..m}) = \Downarrow$ by induction hypothesis. The definition of $\mathfrak{a}^+$ ensures $\mathfrak{a}^+(x, \boldsymbol{s}_{0..m+1}) = \Downarrow$. But $\big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)^{(m+1)}\langle \boldsymbol{s}, x\rangle = \Downarrow$ by definition. Hence, the equation holds in this case.
Finally, assume $\big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)^{(m)}\langle \boldsymbol{s}, x\rangle \neq \Downarrow$ then

$$\big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)^{(m)}\langle \boldsymbol{s}, x\rangle = \langle \boldsymbol{s}_{m..}, \mathfrak{a}^+(x, \boldsymbol{s}_{0..m})\rangle$$

by induction hypothesis i.e. $\mathfrak{a}^+(x, \boldsymbol{s}_{0..m}) \neq \Downarrow$. Thus,

$$\begin{aligned} \big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)^{(m+1)}\langle \boldsymbol{s}, x\rangle &= \big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)\Big(\big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)^{(m)}\langle \boldsymbol{s}, x\rangle\Big) \\ &= \big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)\langle \boldsymbol{s}_{m..}, \mathfrak{a}^+(x, \boldsymbol{s}_{0..m})\rangle. \end{aligned}$$

Now applying (4a), we obtain the required expression for $\big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})\big)^{(m+1)}\langle \boldsymbol{s}, x\rangle$. □

Lemma 5. *For any* $\mathbf{N}$*-detector* $\mathfrak{a}$ *and* $x \in \underline{\mathfrak{a}}$*, the set* $[\![\mathfrak{a}]\!]_x$ *is a safety constraint.*

Proof. Indeed, let us assume the existence of $\boldsymbol{s} \notin [\![\mathfrak{a}]\!]_x$ such that the equation $\boldsymbol{s}'_{0..m} = \boldsymbol{s}_{0..m}$ holds for any $m \in \mathbb{N}$ and for some $\boldsymbol{s}' \in [\![\mathfrak{a}]\!]_x$ depending in generally on m.
The fact $\boldsymbol{s} \notin [\![\mathfrak{a}]\!]_x$ ensures $[\![\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})]\!]\langle \boldsymbol{s}, x\rangle = K$ for some $K \in \mathbb{N}$. It means (see Lemma 4) $\mathfrak{a}^+\big(x, \boldsymbol{s}_{0..K+1}\big) = \Downarrow$. Let $\boldsymbol{s}' \in [\![\mathfrak{a}]\!]_x$ such that $\boldsymbol{s}'_{0..K+1} = \boldsymbol{s}_{0..K+1}$ then $\mathfrak{a}^+\big(x, \boldsymbol{s}'_{0..K+1}\big) = \Downarrow$ and, therefore, $[\![\mathsf{Join}([\boldsymbol{s}'], \mathfrak{a})]\!]\langle \boldsymbol{s}', x\rangle = K$. But $[\![\mathsf{Join}([\boldsymbol{s}'], \mathfrak{a})]\!]\langle \boldsymbol{s}', x\rangle = \infty$ by the assumption $\boldsymbol{s}' \in [\![\mathfrak{a}]\!]_x$. This contradiction completes the proof. □

The following lemma is less simple.

Lemma 6. *For any safety constraint* $S \subset \mathbf{N}^{\mathbb{N}}$*, there exist an* $\mathbf{N}$*-detector* $\mathfrak{a}_S$ *and* $x \in \underline{\mathfrak{a}_S}$ *such that* $[\![\mathfrak{a}_S]\!]_x = S$.

Proof. Let us define the $\mathbf{N}$-detector $\mathfrak{a}_S$ as follows (note that $\boldsymbol{\epsilon} \in \underline{\mathfrak{a}_S}$)

$$\underline{\mathfrak{a}_S} = \{\boldsymbol{\epsilon}\} \bigcup \big\{\boldsymbol{u} \in \mathbf{N}^+ \mid \boldsymbol{u} = \boldsymbol{s}_{0..|\boldsymbol{u}|} \text{ for some } \boldsymbol{s} \in S\big\}$$

$$\mathfrak{a}_S = \lambda\ \boldsymbol{u} \in \underline{\mathfrak{a}_S}\ .\ \lambda\ n \in \mathbf{N}\ .\ \begin{cases} \Downarrow & \text{if } \boldsymbol{u}n \notin \underline{a_S} \\ \boldsymbol{u}n & \text{otherwise} \end{cases}$$

and prove that $[\![\mathfrak{a}_S]\!]_{\boldsymbol{\epsilon}} = S$.
Let us assume $\boldsymbol{s} \in S$ then $\boldsymbol{s}_{0..m} \in \underline{\mathfrak{a}_S}$ for any $m \in \mathbb{N}$. Hence, $\mathfrak{a}_S^+(\boldsymbol{s}_{0..m}, \boldsymbol{s}\,m) = \boldsymbol{s}_{0..m+1} \in \underline{\mathfrak{a}_S}$ for each m i.e. $[\![\mathsf{Join}([\boldsymbol{s}], \mathfrak{a}_S)]\!]\langle \boldsymbol{s}, \boldsymbol{\epsilon}\rangle = \infty$. Thus, $\boldsymbol{s} \in [\![\mathfrak{a}_S]\!]_{\boldsymbol{\epsilon}}$.
Conversely assume $\boldsymbol{s} \in [\![\mathfrak{a}_S]\!]_{\boldsymbol{\epsilon}}$ then $[\![\mathsf{Join}([\boldsymbol{s}], \mathfrak{a}_S)]\!]\langle \boldsymbol{s}, \boldsymbol{\epsilon}\rangle = \infty$ and, therefore, $\big(\mathsf{Join}([\boldsymbol{s}], \mathfrak{a}_S)\big)^{(m)}\langle \boldsymbol{s}, \boldsymbol{\epsilon}\rangle \neq \Downarrow$ for each $m > 0$. Lemma 4 ensures

$$\boldsymbol{s}_{0..m} = \mathfrak{a}_S^+(\boldsymbol{\epsilon}, \boldsymbol{s}_{0..m}) \in \underline{\mathfrak{a}_S} \text{ for all } m > 0$$

that guarantees existence $\boldsymbol{s}^{(m)} \in S$ such that $\boldsymbol{s}^{(m)}_{0..m} = \boldsymbol{s}_{0..m}$. Now using the safeness of S, one can conclude $\boldsymbol{s} \in S$. □

Theorem 4 (about universal detector). *A subset* $S \subset \mathbf{N}^{\mathbb{N}}$ *is a safety constraint if and only if there exist* $P \in \underline{\nu\mathsf{D}_{\mathbf{N}}}$ *such that* $S = [\![\nu\mathsf{D}_{\mathbf{N}}]\!]_P$.

Proof. Lemmas 5 and 6 ensure that a subset $S \subset \mathbf{N}^{\mathbb{N}}$ is a safety constraint if and only if there exist an $\mathbf{N}$-detector $\mathfrak{a}$ and $x \in \underline{\mathfrak{a}}$ such that $S = [\![\mathfrak{a}]\!]_x$. Let us take $P = [\![\mathfrak{a}]\!]x$ and prove $[\![\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})]\!]\langle \boldsymbol{s}, x\rangle = [\![\mathsf{Join}([\boldsymbol{s}], \nu\mathsf{D}_{\mathbf{N}})]\!]\langle \boldsymbol{s}, P\rangle$.
Indeed, for the T-morphism $\mathsf{Join}\left(\mathrm{id}_{\underline{[\boldsymbol{s}]}}, [\![\mathfrak{a}]\!]\right) : \mathsf{Join}([\boldsymbol{s}], \mathfrak{a}) \to \mathsf{Join}([\boldsymbol{s}], \nu\mathsf{D}_{\mathbf{N}})$, we have

$$\Big(\mathsf{Join}\left(\mathrm{id}_{\underline{[\boldsymbol{s}]}}, [\![\mathfrak{a}]\!]\right)\Big)\langle \boldsymbol{s}, x\rangle = \left(\mathrm{id}_{\underline{[\boldsymbol{s}]}} \times [\![\mathfrak{a}]\!]\right)\langle \boldsymbol{s}, x\rangle = \langle \boldsymbol{s}, P\rangle. \tag{5}$$

Using the equation

$$[\![\mathsf{Join}([\boldsymbol{s}], \mathfrak{a})]\!] = [\![\mathsf{Join}([\boldsymbol{s}], \nu\mathsf{D}_{\mathbf{N}})]\!] \circ \Big(\mathsf{Join}\left(\mathrm{id}_{\underline{[\boldsymbol{s}]}}, [\![\mathfrak{a}]\!]\right)\Big)$$

that follows from the definition of an anamorphism, one can derive from (5) the follows

$$[\![\mathsf{Join}([\boldsymbol{s}],\mathfrak{a})]\!]\langle \boldsymbol{s},x\rangle = [\![\mathsf{Join}([\boldsymbol{s}],\nu\mathsf{D}_{\mathbf{N}})]\!]\langle \boldsymbol{s},P\rangle.$$

Considering the last equation holds whenever $P = [\![\mathfrak{a}]\!]x$, one can conclude that $S = [\![\mathfrak{a}]\!]_x = [\![\nu\mathsf{D}_{\mathbf{N}}]\!]_P$. □

Corollary 1. *A subset $S \subset \mathbf{N}^{\mathbb{N}}$ is a safety constraint if and only if it is equal to $\{\boldsymbol{s} \in \mathbf{N}^{\mathbb{N}} \mid \boldsymbol{s}_{0..m} \notin P$ for any $m > 0\}$ for some prefix-free subset of $\mathbf{N}^{+}$.*

Proof. It follows immediately from Theorem 4. □

Corollary 2. *For any $\mathbf{N}$-detectors $\mathfrak{a}$ and $\mathfrak{b}$ and $x \in \underline{\mathfrak{a}}$ and $y \in \underline{\mathfrak{b}}$, the equation $[\![\mathfrak{a}]\!]x = [\![\mathfrak{b}]\!]y$ is sufficient for $[\![\mathfrak{a}]\!]_x = [\![\mathfrak{b}]\!]_y$.*

Proof. The statement is true due to the construction method of P used in the proof of Theorem 4. □

5.3 Families of safety Constraints

Theorem 4 proved in the previous subsection establishes that for the family of all safety constraints, there is a universal detector, i.e. a detector that recognises any safety constraint of the family when the detector configured appropriately. Such a general result, however, cannot be used in the practice of developing software tools, if only because computability considerations limit our expressive capabilities for specifying safety constraints and, in particular, guarantee the impossibility of specifying an arbitrary prefix-free set. Therefore, we need to consider more specific families of safety constraints. In this subsection, we try to outline some general approach to solving this problem.

We begin with the next definition.

Definition 8. *A family of safety constraints $\mathcal{F}$ is below called a **family with a universal detector** if $\mathcal{F} = \{[\![\mathfrak{a}]\!]_x \mid x \in \underline{\mathfrak{a}}\}$ for some $\mathbf{N}$-detector $\mathfrak{a}$ being called in this case a **universal detector** for $\mathcal{F}$.*

Theorem 5. *A family of safety constraints $\mathcal{F}$ is a family with a universal detector if and only if there exists $C \subset \underline{\nu\mathsf{D}_{\mathbf{N}}}$ that meets the following conditions*

$$\begin{aligned} &\textit{for any } P \in C,\\ &\quad \textit{either} \quad n \in P \quad \textit{or} \quad n^{-1}\cdot P \in C \quad \textit{for an arbitrary } n \in \mathbf{N} \end{aligned} \tag{6a}$$

$$\begin{aligned} &\textit{for any safety constraint } S,\\ &\quad S \in \mathcal{F} \textit{ if and only if there exists } P \in C \textit{ such that } S = [\![\nu\mathsf{D}_{\mathbf{N}}]\!]_P \end{aligned} \tag{6b}$$

Proof. Firstly, let us assume the family $\mathcal{F}$ is a family with a universal detector then Prop. 6 guarantees the validity of conditions (6).
Conversely, let us define the next mapping $\mathfrak{a} : C \to \mathsf{D}_{\mathbf{N}}\, C$

$$\mathfrak{a} = \lambda\, P \in C\,.\,\lambda\, n \in \mathbf{N}\,.\,\begin{cases} \Downarrow & \text{if } n \in P \\ n^{-1}\cdot P & \text{otherwise} \end{cases}$$

The correctness of this definition is ensured by (6a), hence $\mathfrak{a}$ is an $\mathbf{N}$-detector and, due to (6b), it is a universal detector. □

Corollary 3. *A family of safety constraints $\mathcal{F}$ is a family with a universal detector if and only if*

$$\{P \in \nu\mathsf{D}_{\mathbf{N}} \mid [\![\nu\mathsf{D}_{\mathbf{N}}]\!]_P \in \mathcal{F}\}$$

is the carrier of a subcoalgebra of $\nu\mathsf{D}_{\mathbf{N}}$.

Now we consider three specific safety constraint family and demonstrate that each of them is a family with a universal detector.

Regular Safety Constraints Here we consider the safety constraint family $\mathcal{R}$ formed as follows a safety constraint S belongs to $\mathcal{R}$ if and only if there exists a detector $\mathfrak{a}$ with the finite carrier such that $S = [\![\mathfrak{a}]\!]_x$ for some $x \in \underline{\mathfrak{a}}$. We call a safety constraint belonging this family a ***regular safety constraint***.

Theorem 6. *The family of regular safety constraints is a family with a universal detector.*

Proof. Let us consider

$$C = \{P \subset \mathbf{N}^+ \mid P = [\![a]\!]x \text{ for some detector } \mathfrak{a} \text{ with finite carrier and } x \in \underline{\mathfrak{a}}\}$$

then any $P \in C$ is a regular prefix-free subset of $\mathbf{N}^+$ (see, for example, [6, Theorem 1]).
Now let us check that C satisfies conditions (6). Indeed, Lemma 2 ensures $n^{-1} \cdot P$ is prefix-free for any $n \in \mathbf{N}$ such that $n \notin P$.
Hence, we need only due to Theorem 5 to check that $n^{-1} \cdot P$ is a regular set under assumption $n \notin P$. To do this we consider a finite Eilenberg machine

$$\langle Q, T \subset Q \times \mathbf{N} \times Q, I \subset Q, F \subset Q \rangle$$

that accepts only words from P (see [5]) and construct the new finite Eilenberg machine

$$\langle Q, T \subset Q \times \mathbf{N} \times Q, I' \subset Q, F \subset Q \rangle$$

where $I' = \{q \in Q \mid \langle q', n, q \rangle \in T \text{ for some } q' \in I\}$. It is evident that the constructed machine accepts exactly $n^{-1} \cdot P$ and, therefore, $n^{-1} \cdot P \in C$. □

Decidable Safety Constraints Here we consider the safety constraint family $\mathcal{D}$ formed as follows a safety constraint S belongs to $\mathcal{D}$ if and only if there exists $P \in \underline{\nu\mathsf{D}_{\mathbf{N}}}$ such that P is decidable and $S = [\![\nu\mathsf{D}_{\mathbf{N}}]\!]_P$. We call a safety constraint belonging this family a ***decidable safety constraint***.

Theorem 7. *The family of decidable safety constraints is a family with a universal detector.*

Proof. To prove the theorem is sufficient to check that the family of a decidable prefix-free subset of $\mathbf{N}^+$ satisfies condition (6a). But this is really true because one can check the statement $\boldsymbol{u} \in n^{-1} \cdot P$ for a decidable prefix-free subset P of $\mathbf{N}^+$, any $\boldsymbol{u} \in \mathbf{N}^+$, and $n \in \mathbf{N}$ such that $n \notin P$, by checking $n\boldsymbol{u} \in P$ with a decision procedure for P. □

Recursively Enumerable Safety Constraints Here we consider the safety constraint family $\mathcal{RE}$ formed as follows a safety constraint S belongs to $\mathcal{RE}$ if and only if there exists $P \in \overline{\boldsymbol{\nu}\mathsf{D}_{\mathbf{N}}}$ such that P is recursively enumerable and $S = [\![\boldsymbol{\nu}\mathsf{D}_{\mathbf{N}}]\!]_P$. We call a safety constraint belonging this family a ***recursively enumerable safety constraint***.

Theorem 8. *The family of recursively enumerable safety constraints is a family with a universal detector.*

Proof. Similarly to proof of the previous theorem, we need to furnish a semi-decision procedure for checking the statement $\boldsymbol{u} \in n^{-1} \cdot P$ with a recursively enumerable prefix-free subset P of $\mathbf{N}^+$, any $\boldsymbol{u} \in \mathbf{N}^+$, and $n \in \mathbf{N}$ such that $n \notin P$. This procedure is as follows
(1) run parallelly checkings $n\boldsymbol{u}_{0..k} \in P$ for $k = 1, \ldots, |\boldsymbol{u}|$;
(2) wait for any of these runs to halt;
(3) report success if the halting run is the run for checking $n\boldsymbol{u} \in P$.
An event waited in item (2) may do not happen if $n\boldsymbol{u} \notin P$ i.e. $\boldsymbol{u} \notin n^{-1} \cdot P$.
If the wait in paragraph (2) is completed then only for one $0 < k \leq |\boldsymbol{u}|$ the corresponding checking is completed due to P is prefix-free.
$\boldsymbol{u} \in n^{-1} \cdot P$ or equivalently $n\boldsymbol{u} \in P$ if and only if the checking with number $|\boldsymbol{u}| - 1$ is halted.
In other cases, $n\boldsymbol{u} \notin P$ i.e. $\boldsymbol{u} \notin n^{-1} \cdot P$.
Thus, the furnished procedure is really a semi-decision procedure for the statement $\boldsymbol{u} \in n^{-1} \cdot P$. □

6 Conclusion

Summing up the mentioned above we can conclude that subdetectors of a final detector correspond to families of safety constraints with universal detectors. These families are candidates for semantic models of domain-specific languages for specification of safety constraints.

Of course, the obtained results tell nothing how to construct such languages. But they turn this problem into more precisely defined.

We can identify as problems for further studying the follows
(1) Give syntactic characterisation of regular safety constraints family.
(2) Understand what classes of grammars (for example, context-free) define proper families of safety constraints.
(3) Can we use complexity classes for defining families of safety constraints with universal detectors?
(4) Understand how compositional theory of detectors can be developed.

An especial area of further research is the dissemination of the proposed approach for the specification and analysis of causality constraints, formulated in terms of the logical clock model. Now authors are working on the paper "Understanding Logical Clock Model Coalgebraically".

References

1. Aczel, P., Mendler, N.: A final coalgebra theorem. In: Proc. CTCS'89, Lecture Notes in Comput. Sci., vol. 389, pp. 357–365. Springer (1989)
2. Alpern, B., Schneider, F.: Recognizing safety and liveness. Distributed Computing **2**, 117–126 (1987)
3. Awodey, S.: Category Theory. Oxford University Press, 2nd edn. (2010)
4. Bowen, J., Stavridou, V.: Safety-critical systems, formal methods and standards. Software Engineering Journal **8**(4), 189–209 (1993)
5. Eilenberg, S.: Automata, Languages and Machines, vol. A. Academic Press (1974)
6. G. Zholtkevych and N. Polyakovska: Machine Learning Technique for Regular Pattern Detector Synthesis: toward Mathematical Rationale. In: Computational Linguistics and Intelligent Systems, CEUR Workshop Proceedings, vol. 2362, pp. 254–265. CEUR-WS (2019)
7. Grant, E.S.: Requirements engineering for safety critical systems: An approach for avionic systems. In: 2nd IEEE International Conference on Computer and Communications (ICCC). pp. 991–995. IEEE (2016)
8. Jacobs, B.: Objects And Classes, Co-Algebraically. In: Freitag, B., et al. (eds.) Object Orientation with Parallelism and Persistence, The Springer International Series in Engineering and Computer Science, vol. 370, pp. 83–103. Springer, Boston, MA (1996)
9. Jacobs, B.: Introduction to Coalgebra: Towards Mathematics of States and Observation. Cambridge Tracts in Theoretical Computer Science, Cambridge University Press (2017)
10. Lane, S.M.: Categories for the Working Mathematician. Springer, 2nd edn. (2010)
11. Lee, E.A.: Cyber Physical Systems: Design Challenges. Tech. Rep. UCB/EECS-2008-8, EECS Department, University of California, Berkeley (Jan 2008), http://www2.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-8.html
12. Mana, Z., Pnueli, A.: The Temporal Logic of Reactive and Concurrent Systems: Specifications. Springer–Verlag (1992)
13. Rutten, J.: Universal coalgebra: a theory of systems. Theor. Comput. Sci. **249**, 3–80 (2000)