

# Paraconsistent Transition Systems\*

Ana Cruz

INESC TEC, University of Minho, Portugal

Alexandre Madeira

CIDMA, University of Aveiro, Portugal

Luís S. Barbosa

INESC TEC, University of Minho, Portugal

Often in Software Engineering a modelling formalism has to support scenarios of inconsistency in which several requirements either reinforce or contradict each other. Paraconsistent transition systems are proposed in this paper as one such formalism: states evolve through two accessibility relations capturing weighted evidence of a transition or its absence, respectively. Their weights come from a specific residuated lattice. A category of these systems, and the corresponding algebra, is defined providing a formal setting to model different application scenarios. One of them, dealing with the effect of quantum decoherence in quantum programs, is used for illustration purposes.

## 1 Introduction

Dealing with application scenarios where requirements either reinforce or contradict each other is not uncommon in Software Engineering. One such scenarios comes from current practice in quantum computation in the context of NISQ (*Noisy Intermediate-Scale Quantum*) technology [11] in which levels of decoherence of quantum memory need to be articulated with the length of the circuits to assess program quality.

In a recent paper [7], the authors introduced a new kind of weighted transitions systems which records, for each transition, a positive and negative weight which, informally, capture the degree of effectiveness (*'presence'*) and of impossibility (*'absence'*) of a transition. This allows the model to capture both *vagueness*, whenever both weights sum less than 1, as usual e.g. in fuzzy systems, and *inconsistency*, when their sum exceeds 1. This last feature motivates the qualifier *paraconsistent* borrowed from the work on paraconsistent logic [9, 5], which accommodates inconsistency in a controlled way, treating inconsistent information as potentially informative. Such logics were originally developed in Latin America in the decades of 1950 and 1960, mainly by F. Asenjo and Newton da Costa. Quickly, however, the topic attracted attention in the international community and the original scope of mathematical applications broadened out, as witnessed in a recent book emphasizing the engineering potential of paraconsistency [2]. In particular, a number of applications to themes from quantum mechanics and quantum information theory have been studied by D. Chiara [4] and W. Carnielli and his collaborators [1, 6].

This paper continues such a research program in two directions. First it introduces a suitable notion of morphism for paraconsistent labelled transition systems (PLTS) leading to the definition of the corresponding category and its algebra. Notions of simulation, bisimulation and trace for PLTS are also discussed. On a second direction, the paper discusses an application of PLTS to reason about the effect of quantum decoherence in quantum programs.

---

\*This work is supported by by FCT, the Portuguese funding agency for Science and Technology with the projects UIDB/04106/2020 and PTDC/CCI-COM/4280/2021

**Paper structure.** After recalling the concept of a PLTS and defining their morphisms in section 2, section 3 discusses suitable notions of simulation, bisimulation and trace. Compositional construction of (pointed) PLTS are characterised in section 4 by exploring the relevant category, following G. Winskel and M. Nielsen’s ‘recipe’ [13]. Section 5 illustrates their use to express quantum circuits with decoherence. Finally, section 6 concludes and points out a number of future research directions.

## 2 Paraconsistent labelled transition systems

A *paraconsistent labelled transition system* (PLTS) incorporates two accessibility relations, classified as positive and negative, respectively, which characterise each transition in opposite ways: one represents the evidence of its presence and other the evidence of its absence. Both relations are weighted by elements of a residuated lattice  $\Sigma = \langle \wedge, \vee, \odot, \rightarrow, 1, 0 \rangle$ , where,  $\langle A, \wedge, \vee, 1, 0 \rangle$  is a lattice,  $\langle A, \odot, 1 \rangle$  is a monoid, and operation  $\odot$  is residuated, with  $\rightarrow$ , i.e. for all  $a, b, c \in A$ ,  $a \odot b \leq c \Leftrightarrow b \leq a \rightarrow c$ . A Gödel algebra  $G = \langle [0, 1], \min, \max, \min, \rightarrow, 0, 1 \rangle$  is an example of such a structure, that will be used in the sequel. Operators *max* and *min* retain the usual definitions, whereas implication is given by

$$a \rightarrow b = \begin{cases} 1, & \text{if } a \leq b \\ b, & \text{otherwise} \end{cases}.$$

Our constructions, however, are, to a large extent, independent of the particular residuated lattice chosen. The definition below extends the one in reference [7] to consider labels in an explicit way. Thus,

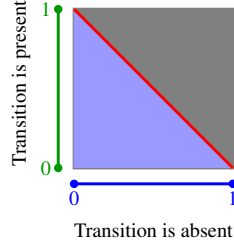
**Definition 1.** A *paraconsistent labelled transition system* (PLTS) over a residuated lattice  $A$  and a set of atomic actions  $\Pi$  is a structure  $\langle W, R, \Pi \rangle$  where,  $W$  is a non-empty set of states,  $\Pi$  is a set of labels, and  $R \subseteq W \times \Pi \times W \times A \times A$  characterises its dynamics, subjected to the following condition: between two arbitrary states there is at most one transition involving label  $a$ , for every  $a \in \Pi$ . Each tuple  $(w_1, a, w_2, \alpha, \beta) \in R$  represents a transition from  $w_1$  to  $w_2$  labelled by  $(a, \alpha, \beta)$ , where  $\alpha$  is the degree to which the action  $a$  contributes to a transition from  $w_1$  to  $w_2$ , and  $\beta$ , dually, expresses the degree to which it prevents its occurrence.

The condition imposed in the definition above makes it possible to express relation  $R$  in terms of a positive and a negative accessibility relation  $r^+, r^- : \Pi \longrightarrow A^{W \times W}$ , with

$$r^+(\pi)(w, w') = \begin{cases} \alpha & \text{if } (w, \pi, w', \alpha, \beta) \in R \\ 0 & \text{otherwise} \end{cases}$$

and  $r^-$  defined similarly. These two relations jointly express different behaviours associated to a transition:

- *inconsistency*, when the positive and negative weights are contradictory, i.e. they sum to some value greater than 1; this corresponds to the upper triangle in the picture below, filled in grey.
- *vagueness*, when the sum is less than 1, corresponding to the lower, periwinkle triangle in the same picture;
- *consistency*, when the sum is exactly 1, which means that the measures of the factors enforcing or preventing a transition are complementary, corresponding to the red line in the picture.

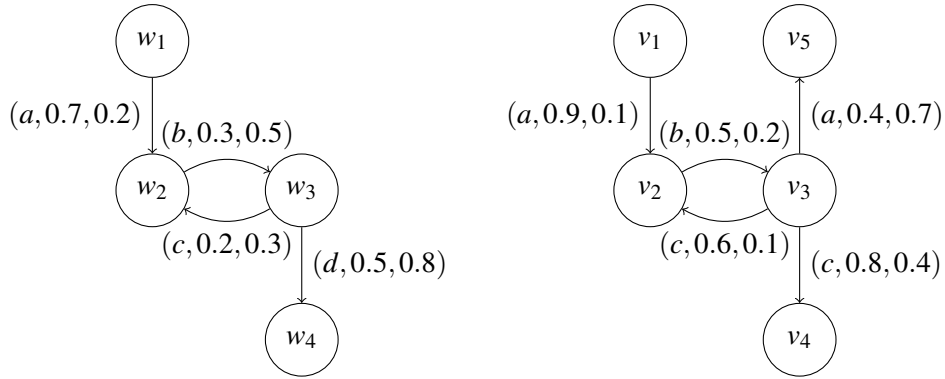


Morphisms between PLTS respect, as one would expect, the structure of both accessibility relations. Formally,

**Definition 2.** Let  $T_1 = \langle W_1, R_1, \Pi \rangle$ ,  $T_2 = \langle W_2, R_2, \Pi \rangle$  be two PLTSs defined over the same set of actions  $\Pi$ . A *morphism* from  $T_1$  to  $T_2$  is a function  $h : W_1 \rightarrow W_2$  such that

$$\forall a \in \Pi, r_1^+(a)(w_1, w_2) \leq r_2^+(a)(hw_1, hw_2) \text{ and } r_1^-(a)(w_1, w_2) \geq r_2^-(a)(hw_1, hw_2)$$

**Example 1.** Function  $h = \{w_1 \mapsto v_1, w_2 \mapsto v_2, w_3 \mapsto v_3\}$  is a morphism from  $M_1$  to  $M_2$ , over  $\Pi = \{a, b, c, d\}$ , depicted below



### 3 Simulation and Bisimulation for PLTS

Clearly, PLTSs and their morphisms form a category, with composition and identities borrowed from Set. To compare PLTSs is also useful to define what simulation and bisimulation mean in this setting. Thus, under the same assumptions on  $T_1$  and  $T_2$ ,

**Definition 3.** A relation  $S \subseteq W_1 \times W_2$  is a *simulation* provided that, for all  $\langle p, q \rangle \in S$  and  $a \in \Pi$ ,

$$p \xrightarrow{(a, \alpha, \beta)}_{T_1} p' \Rightarrow \langle \exists q' \in W_2 \cdot \exists \gamma, \delta \in [0, 1] \cdot q \xrightarrow{(a, \gamma, \delta)}_{T_2} q' \wedge \langle p', q' \rangle \in S \wedge \gamma \geq \alpha \wedge \delta \leq \beta \rangle$$

which can be abbreviated to

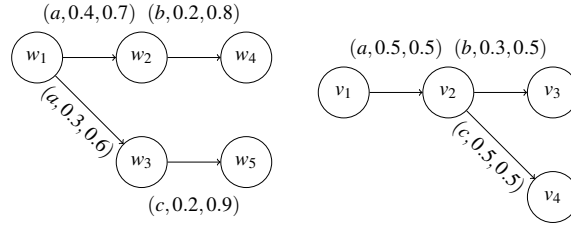
$$p \xrightarrow{(a, \alpha, \beta)}_{T_1} p' \Rightarrow \langle \exists q' \in W_2 \cdot q \xrightarrow{(a, \gamma: \gamma \geq \alpha, \delta: \delta \leq \beta)}_{T_2} q' \wedge \langle p', q' \rangle \in S \rangle$$

Two states  $p$  and  $q$  are *similar*, written  $p \lesssim q$ , if there is a simulation  $S$  such that  $\langle p, q \rangle \in S$ .

Whenever one restricts in the definition above to the existence of values  $\gamma$  (resp.  $\delta$ ) such that  $\gamma \geq \alpha$  (resp.  $\delta \leq \beta$ ), the corresponding simulation is called *positive* (resp. *negative*).

**Example 2.** In the PLTSs depicted below,  $w_1 \lesssim v_1$ , witnessed by

$$S = \{ \langle w_1, v_1 \rangle, \langle w_2, v_2 \rangle, \langle w_3, v_2 \rangle, \langle w_4, v_3 \rangle, \langle w_5, v_4 \rangle \}$$



Finally,

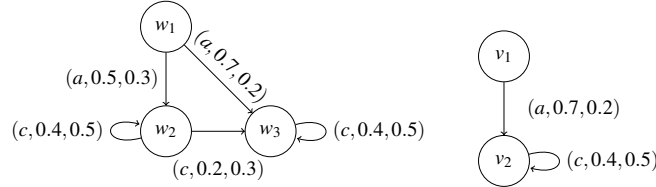
**Definition 4.** A relation  $B \subseteq W_1 \times W_2$  is a **bisimulation** if for  $\langle p, q \rangle \in B$  and  $a \in \Pi$

$$p \xrightarrow{(a, \alpha, \beta)}_{M_1} p' \Rightarrow \langle \exists q' \in W_2 : q \xrightarrow{(a, \alpha, \beta)}_{M_2} q' \wedge \langle p', q' \rangle \in B \rangle$$

$$q \xrightarrow{(a, \alpha, \beta)}_{M_2} q' \Rightarrow \langle \exists p' \in W_1 : p \xrightarrow{(a, \alpha, \beta)}_{M_1} p' \wedge \langle p', q' \rangle \in B \rangle$$

Two states  $p$  and  $q$  are **bisimilar**, written  $p \sim q$ , if there is a bisimulation  $B$  such that  $\langle p, q \rangle \in B$ .

**Example 3.** Consider the two PLTSs depicted below. Clearly,  $w_1 \sim v_1$ .



**Lemma 1.** Similarity,  $\lesssim$ , and bisimilarity,  $\sim$ , form a preorder and an equivalence relation, respectively.

*Proof.* The proof is similar to one for classical labelled transition systems (details in [8]).  $\square$

As usual, a *trace* from a given state  $w$  in a PLTS  $T$  is simply the sequence  $s$  of tuples  $(a, \alpha, \beta)$  labelling a path in  $T$  starting at  $w$ . A first projection on such a sequence, i.e.  $\pi_1^*(s)$  retrieves the corresponding sequence of labels that constitutes what may be called an *unweighted trace*. More interesting is the notion of *weighted trace* which appends to the sequence of labels, the maximum value for the positive accessibility relation and the minimum value for the negative accessibility relation computed along the trace  $s$ . Formally,

**Definition 5.** Given a trace  $s$  in a PLTS  $T$ , the corresponding weighted trace is defined by

$$tw(s) = \langle \pi_1^*, \bigwedge (\pi_2^*), \bigvee (\pi_3^*) \rangle (s)$$

where,  $\pi_n$  denotes the  $n$  projection in a tuple,  $\langle f, g, h \rangle$  is the universal arrow to a Cartesian product,  $f^*$  is the functorial extension of  $f$  to sequences over its domain, and  $\bigwedge$  (resp.  $\bigvee$ ) are the distributed version of  $\wedge$  (resp.  $\vee$ ) over sequences.

**Definition 6.** A weighted trace  $t = \langle [a_1, a_2, \dots, a_m], \alpha, \beta \rangle$  is a **weighted subtrace** of  $t' = \langle [b_1, b_2, \dots, b_n], \gamma, \delta \rangle$  if (i) sequence  $[a_1, a_2, \dots, a_m]$  is a prefix of  $[b_1, b_2, \dots, b_n]$ , (ii)  $\gamma \geq \alpha$  and (iii)  $\delta \leq \beta$ . The definition lifts to sets as follows: given two sets  $X$  and  $Y$  of weighted traces,

$$X \sqsubseteq Y \text{ iff } \forall t \in X. \exists t' \in Y. t \text{ is a weighted subtrace of } t'$$

**Example 4.** Consider again the two PLTSs given in Example 2. The weighted traces from  $w_1$  are  $\{t_1 = \langle [a, b], 0.2, 0.8 \rangle, t_2 = \langle [a, c], 0.2, 0.9 \rangle\}$  and the ones from  $v_1$  are  $\{t'_1 = \langle [a, b], 0.5, 0.5 \rangle, t'_2 = \langle [a, c], 0.5, 0.5 \rangle\}$ . Clearly,  $t_1$  (resp.  $t_2$ ) is a weighted subtrace of  $t'_1$  (resp.  $t'_2$ ).

**Lemma 2.** Consider two PLTSs,  $T_1 = \langle W_1, R_1 \rangle$  and  $T_2 = \langle W_2, R_2 \rangle$ . If two states  $p \in W_1$  and  $q \in W_2$  are similar (resp. bisimilar), i.e.,  $p \lesssim q$  (resp.  $p \sim q$ ), then the set of weighted traces from  $p$ ,  $X$ , and the set of weighted traces from  $q$ ,  $Y$ , are such that  $X \sqsubseteq Y$  (resp. coincide).

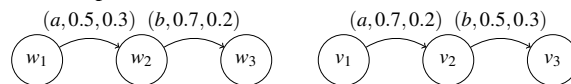
*Proof.* If  $p \lesssim q$  each trace  $t$  from  $p$  is a prefix of trace  $t'$  from  $q$ . Let  $[\alpha_1, \alpha_2, \dots, \alpha_m]$  and  $[\beta_1, \beta_2, \dots, \beta_m]$  be the sequences of positive and negative weights associated to  $t$ . Similarly, let  $[\alpha'_1, \alpha'_2, \dots, \alpha'_n]$  and  $[\beta'_1, \beta'_2, \dots, \beta'_n]$  be the corresponding sequences for  $t'$ ; of course  $m \leq n$ . As  $(p, q)$  belongs to a simulation,  $\alpha'_i \geq \alpha_i$  and  $\beta'_i \leq \beta_i$ , for all  $i \leq n$ . So,  $\text{Min}[\alpha'_1, \alpha'_2, \dots, \alpha'_m] \geq \text{Min}[\alpha_1, \alpha_2, \dots, \alpha_m]$  and  $\text{Max}[\alpha'_1, \alpha'_2, \dots, \alpha'_m] \leq \text{Max}[\alpha_1, \alpha_2, \dots, \alpha_m]$ . Note that  $\text{Min}$  and  $\text{Max}$  correspond to  $\wedge$  and  $\vee$  in a Gödel algebra. Thus,

$$\langle t, \text{Min}[\alpha_1, \alpha_2, \dots, \alpha_m], \text{Max}[\alpha_1, \alpha_2, \dots, \alpha_m] \rangle$$

is a weighted subtrace of  $\langle t'|_m, \text{Min}[\alpha'_1, \alpha'_2, \dots, \alpha'_m], \text{Max}[\alpha'_1, \alpha'_2, \dots, \alpha'_m] \rangle$ , where  $t'|_m$  is the subsequence of  $t'$  with  $m$  elements. The statement for  $\sim$  follows similarly.  $\square$

Note that the converse of this lemma does not hold, as shown by the following counterexample.

**Example 5.** Consider the PLTS depicted below.



$X = \{\langle [a], 0.5, 0.3 \rangle, \langle [a, b], 0.5, 0.3 \rangle\}$  is the set of weighted traces from  $w_1$ . Similarly,  $Y = \{\langle [a], 0.7, 0.2 \rangle, \langle [a, b], 0.5, 0.3 \rangle\}$  is the corresponding set from  $w_2$ . Clearly  $\langle [a], 0.5, 0.3 \rangle$  is a weighted subtrace of  $\langle [a], 0.7, 0.2 \rangle$ . Thus  $X \sqsubseteq Y$ . However,  $w_1 \not\lesssim w_2$ .

## 4 New PLTS from old

New PLTS can be built compositionally. This section introduces the relevant operators by exploring the structure of the category of Pt of *pointed* PLTS, i.e. whose objects are PLTSs with a distinguished initial state, i.e.  $\langle W, i, R, \Pi \rangle$ , where  $\langle W, R, \Pi \rangle$  is a PLTS and  $i \in W$ . Arrows in Pt are allowed between PLTSs with different sets of labels, therefore generalizing Definition 2 as follows:

**Definition 7.** Let  $T_1 = \langle W_1, i_1, R_1, \Pi \rangle$  and  $T_2 = \langle W_2, i_2, R_2, \Pi' \rangle$  be two pointed PLTSs. A morphism in Pt from  $T_1$  to  $T_2$  is a pair of functions  $(\sigma : W_1 \rightarrow W_2, \lambda : \Pi \rightarrow_{\perp} \Pi')$  such that<sup>1</sup>  $\sigma(i_1) = i_2$ , and, if  $(w, a, w', \alpha, \beta) \in R_1$  then  $(\sigma(w), \lambda(a), \sigma(w'), \alpha', \beta') \in R_2^{\perp}$ , with  $\alpha \leq \alpha'$  and  $\beta' \leq \beta$ , where, for an accessibility relation  $R$ ,  $R^{\perp} = R \cup \{(w, \perp, w, 1, 0) \mid w \in W\}$  denotes  $R$  enriched with idle transitions in each state.

Clearly Pt forms a category, with composition inherited from Set and  $\text{Set}_{\perp}$ , the later standing for the category of sets and partial functions, with  $T_{\text{nil}} = \langle \{*\}, *, \emptyset, \emptyset \rangle$  as both the initial and final object. The corresponding unique morphisms are  $! : T \rightarrow T_{\text{nil}}$ , given by  $\langle *, () \rangle$ , and  $? : T_{\text{nil}} \rightarrow T$ , given by  $\langle i, () \rangle$ , where  $()$  is the empty map and notation  $\underline{x}$  stands for the constant, everywhere  $x$ , function.

An algebra of PLTS typically includes some form of parallel composition, disjoint union, restriction, relabelling and prefixing, as one is used from the process algebra literature [3]. Accordingly, these operators are defined along the lines proposed by G. Winskel and M. Nielsen [13], for the standard, more usual case.

<sup>1</sup>Notation  $\lambda : \Pi \rightarrow_{\perp} \Pi'$  stands for the totalization of a partial function by mapping to  $\perp$  all elements of  $\Pi$  for which the function is undefined.

**Restriction.** The restriction operator is intended to control the interface of a transition system, preserving, in the case of a PLTS, the corresponding positive and negative weights. Formally,

**Definition 8.** Let  $T = \langle W, i, R, \Pi \rangle$  be a PLTS, and  $\lambda : \Pi' \rightarrow \Pi$  be an inclusion. The **restriction** of  $T$  to  $\lambda$ ,  $T \upharpoonright \lambda$ , is a PLTS  $\langle W, i, R', \Pi' \rangle$  over  $\Pi'$  such that  $R' = \{(w, \pi, w', \alpha, \beta) \in R \mid \pi \in \Pi'\}$ .

There is a morphism  $f = (1_W, \lambda)$  from  $T \upharpoonright \lambda$  to  $T$ , and a functor  $P : \text{Pt} \rightarrow \text{Set}_\perp$  which sends a morphism  $(\sigma, \lambda) : T \rightarrow T'$  to the partial function  $\lambda : \Pi' \rightarrow \Pi$ . Clearly,  $f$  is the Cartesian lifting of morphism  $P(f) = \lambda$  in  $\text{Set}_\perp$ . Being Cartesian means that for any  $g : T' \rightarrow T$  in  $\text{Pt}$  such that  $P(g) = \lambda$  there is a unique morphism  $h$  such that  $P(h) = 1_{\Pi'}$  making the following diagram to commute:

$$\begin{array}{ccc} T' & & \\ h \downarrow & \searrow g & \\ T \upharpoonright \lambda & \xrightarrow{f} & T \end{array}$$

Note that, in general, restriction does not preserve reachable states. Often, thus, the result of a restriction is itself restricted to its reachable part.

**Relabelling.** In the same group of *interface-modifier* operators, is *relabelling*, which renames the labels of a PLTS according to a total function  $\lambda : \Pi \rightarrow \Pi'$ .

**Definition 9.** Let  $T = \langle W, i, R, \Pi \rangle$  be a PLTS, and  $\lambda : \Pi \rightarrow \Pi'$  be a total function. The **relabelling** of  $T$  according to  $\lambda$ ,  $T\{\lambda\}$  is the PLTS  $\langle W, i, R', \Pi' \rangle$  where  $R' = \{(w, \lambda(a), w', \alpha, \beta) \mid (w, a, w', \alpha, \beta) \in R\}$ .

Dually to the previous case, there is a morphism  $f = (1_W, \lambda)$  from  $T$  to  $T\{\lambda\}$  which is the cocartesian lifting of  $\lambda$  ( $= P(f)$ ).

**Parallel composition.** The product of two PLTSs combines their state spaces and includes all *synchronous* transitions, triggered by the simultaneous occurrence of an action of each component, as well as *asynchronous* ones in which a transition in one component is paired with an *idle* transition, labelled by  $\perp$ , in the other. Formally,

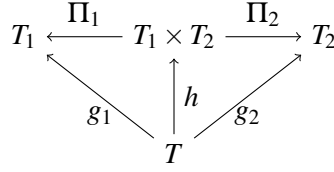
**Definition 10.** Let  $T_1 = \langle W_1, i_1, R_1, \Pi_1 \rangle$  and  $T_2 = \langle W_2, i_2, R_2, \Pi_2 \rangle$  be two PLTS. Their **parallel composition**  $T_1 \times T_2$  is the PLTS  $\langle W_1 \times W_2, (i_1, i_2), R, \Pi' \rangle$ , such that  $\Pi' = \Pi_1 \times_\perp \Pi_2 = \{(a, \perp) \mid a \in \Pi_1\} \cup \{(\perp, b) \mid b \in \Pi_2\} \cup \{(a, b) \mid a \in \Pi_1, b \in \Pi_2\}$ , and  $(w, a, w', \alpha, \beta) \in R$  if and only if  $(\pi_1(w), \pi_1(a), \pi_1(w'), \alpha_1, \beta_1) \in R_1^\perp$ ,  $(\pi_2(w), \pi_2(a), \pi_2(w'), \alpha_2, \beta_2) \in R_2^\perp$ ,  $\alpha = \min(\alpha_1, \alpha_2)$  and  $\beta = \max(\beta_1, \beta_2)$ .

**Lemma 3.** *Parallel composition is the product construction in Pt.*

*Proof.* In the diagram below let  $g_i = (\sigma_i, \lambda_i)$ , for  $i = 1, 2$ , and define  $h$  as  $h = (\langle \sigma_1, \sigma_2 \rangle, \langle \lambda_1, \lambda_2 \rangle)$ , where  $\langle f_1, f_2 \rangle(x) = (f_1(x), f_2(x))$  is the universal arrow in a product diagram in  $\text{Set}$ . Clearly,  $h$  lifts universality to  $\text{Pt}$ , as the unique arrow making the diagram to commute. It remains show it is indeed an arrow in the category. Indeed, let  $T = \langle W, i, R, \Pi \rangle$ ,  $T_1 = \langle W_1, i_1, R_1, \Pi_1 \rangle$ , and define  $T_1 \times T_2 = \langle W_1 \times W_2, (i_1, i_2), R', \Pi' \rangle$  according to definition 10. Thus, for each  $(w, a, w', \alpha, \beta) \in R$ , there is a transition  $(\sigma_1(w), \lambda_1(a), \sigma_1(w'), \alpha_1, \beta_1) \in R_1^\perp$  such that  $\alpha \leq \alpha_1$  and  $\beta \geq \beta_1$ ; and also a transition  $(\sigma_2(w), \lambda_2(a), \sigma_2(w'), \alpha_2, \beta_2) \in R_2^\perp$  such that  $\alpha \leq \alpha_1$  and  $\beta \geq \beta_2$ . Moreover, there is a transition

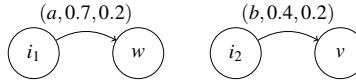
$$(\langle \sigma_1, \sigma_2 \rangle(w), \langle \lambda_1, \lambda_2 \rangle(a), \langle \sigma_1, \sigma_2 \rangle(w'), \min(\alpha_1, \alpha_2), \max(\beta_1, \beta_2)) \in R'$$

Thus, there is a transition  $(\langle \sigma_1, \sigma_2 \rangle(w), \langle \lambda_1, \lambda_2 \rangle(a), \langle \sigma_1, \sigma_2 \rangle(w'), \alpha', \beta')$   $\in R'$ , for any  $(w, a, w', \alpha, \beta) \in R$ , such that  $\alpha \leq \alpha'$  and  $\beta \geq \beta'$ . Furthermore,  $\langle \sigma_1, \sigma_2 \rangle(i) = (\sigma_1(i), \sigma_2(i)) = (i_1, i_2)$ . This establishes  $h$  as a  $\text{Pt}$  morphism.

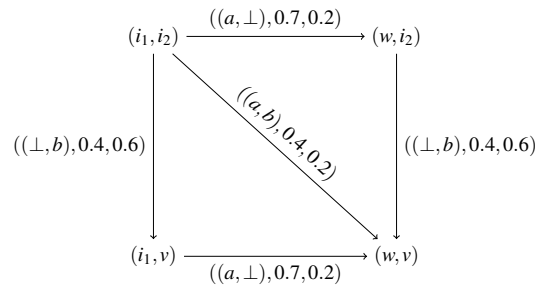


□

**Example 6.** Consider the two PLTSs,  $T_1$  and  $T_2$ , depicted below.



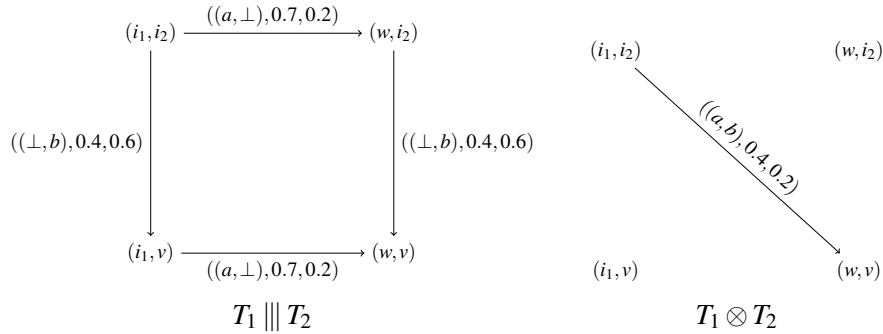
Their product  $T$  is the PLTS



A suitable combination of parallel composition and restriction may enforce different synchronization disciplines. For example, *interleaving* or *asynchronous product*  $T_1 \parallel T_2$  is defined as  $(T_1 \times T_2) \upharpoonright \lambda$  with the inclusion  $\lambda : \Pi \rightarrow \Pi_1 \times_{\perp} \Pi_2$  for  $\Pi = \{(a, \perp) \mid a \in \Pi_1\} \cup \{(\perp, b) \mid b \in \Pi_2\}$ . This results in a PLTS  $\langle W_1 \times W_2, (i_1, i_2), R, \Pi \rangle$  such that  $R = \{(w, a, w', \alpha, \beta) \in R' \mid a \in \Pi\}$ .

Similarly, the *synchronous product*  $T_1 \otimes T_2$  is also defined as  $(T_1 \times T_2) \upharpoonright \lambda$ , taking now  $\Pi = \{(a, b) \mid a \in \Pi_1 \text{ and } b \in \Pi_2\}$  as the domain of  $\lambda$ .

**Example 7.** Interleaving and synchronous product of  $T_1$  and  $T_2$  as in Example 8, are depicted below.



**Sum.** The sum of two PLTSs corresponds to their non-deterministic composition: the resulting PLTS behaves as either of its components. Formally,

**Definition 11.** Let  $T_1 = \langle W_1, i_1, R_1, \Pi_1 \rangle$  and  $T_2 = \langle W_2, i_2, R_2, \Pi_2 \rangle$  be two PLTSs. Their sum  $T_1 + T_2$  is the PLTS  $\langle W, (i_1, i_2), R, \Pi_1 \cup \Pi_2 \rangle$ , where

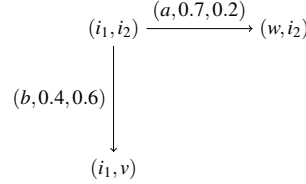
$$- W = (W_1 \times \{i_2\}) \cup (\{i_1\} \times W_2),$$

- $t \in R$  if and only if there exists a transition  $(w, a, w', \alpha, \beta) \in R_1$  such that  $t = (\iota_1(w), a, \iota_1(w'), \alpha, \beta)$ , or a transition  $(w, a, w', \alpha, \beta) \in R_2$  such that  $t = (\iota_2(w), a, \iota_2(w'), \alpha, \beta)$

where  $\iota_1$  and  $\iota_2$  are the left and right injections associated to a coproduct in  $\text{Set}$ , respectively.

Sum is actually a coproduct in  $\text{Pt}$  (the proof follows the argument used for the product case), making  $T_1 + T_2$  dual to  $T_1 \times T_2$ .

**Example 8.** The sum  $T_1 + T_2$ , for  $T_1, T_2$  defined as in Example 8 is given by



**Prefixing.** As a limited form of sequential composition, prefix appends to a pointed PLTS a new initial state and a new transition to the previous initial state, after which the system behaves as the original one.

**Definition 12.** Let  $T = \langle W, i, R, \Pi \rangle$  be a PLTS and  $w_{new}$  a fresh state identifier not in  $W$ . Given an action  $a$ , and  $\alpha, \beta \in [0, 1]$ , the prefix  $(a, \alpha, \beta)T$  is defined as  $\langle W \cup \{w_{new}\}, w_{new}, R', \Pi \cup \{a\} \rangle$  where  $R' = R \cup (w_{new}, a, i, \alpha, \beta)$ .

Since it is not required that the prefixing label is distinct from the ones in the original system, prefixing does not extend to a functor in  $\text{Pt}$ , as illustrated in the counterexample below. This is obviously the case for a category of classical labelled transition systems as well. In both cases, however, prefix extends to a functor if the corresponding categories are restricted to action-preserving morphisms, i.e. in which the action component of a morphism is always an inclusion

**Example 9.** Consider two pointed PLTS  $T_1$  and  $T_2$



connected by a morphism  $(\sigma, \lambda) : T_1 \rightarrow T_2$  such that  $\sigma(i_1) = i_2$ ,  $\sigma(w) = v$  and  $\lambda(a) = b$ . Now consider the prefixes  $(a, 1, 0)T_1$  and  $(a, 1, 0)T_2$  depicted below.



Clearly, a mapping from the actions in  $(a, 1, 0)T_1$  to the actions in  $(a, 1, 0)T_2$  does not exist so neither exists a morphism between the two systems.

**Functorial extensions.** Other useful operations between PLTSs, typically acting on transitions' positive and negative weights, and often restricted to PLTSs over a specific residuated lattice, can be defined functorially in  $\text{Pt}$ . An example involving a PLTS defined over a Gödel algebra is an operation that uniformly increases or decreases the value of the positive (or the negative, or both) weight in all transitions. Let

$$a \oplus b = \begin{cases} 1 & \text{if } a + b \geq 1 \\ 0 & \text{if } a + b \leq 0 \\ a + b & \text{otherwise} \end{cases}$$

Thus,



**Definition 13.** Let  $T = \langle W, i, R, \Pi \rangle$  be a PLTS. Taking  $v \in [-1, 1]$ , the **positive  $v$ -approximation**  $T_{\oplus v}^+$  is a PLTS  $\langle W, i, R', \Pi \rangle$  where

$$R' = \{(w, \pi, w', \alpha \oplus v, \beta) \mid (w, \pi, w', \alpha, \beta) \in R\}.$$

The definition extends to a functor in Pt which is the identity in morphisms. Similar operations can be defined to act on the negative accessibility relation or both.

Another useful operation removes all transitions in a pointed PLTS for which the positive accessibility relation is below a certain value and the negative accessibility relation is above a certain value. Formally,

**Definition 14.** Let  $T = \langle W, i, R, \Pi \rangle$  be a pointed PLTS, and  $p, n \in [0, 1]$ . The **purged PLTS**  $T_{p\uparrow, n\downarrow}$  is defined as  $\langle W, i, R', \Pi \rangle$  where

$$R' = \{(w, \pi, w', \alpha, \beta) \mid (w, \pi, w', \alpha, \beta) \in R \text{ and } \alpha \geq p \text{ and } \beta \leq m\}$$

Clearly, the operation extends to a functor in Pt, mapping morphisms to themselves.

## 5 An application to quantum circuit optimization

In a quantum circuit [10] decoherence consists in decay of a qubit in superposition to its ground state and may be caused by distinct physical phenomena. A quantum circuit is effective only if gate operations and measurements are performed to superposition states within a limited period of time after their preparation. In this section pointed PLTS will be used to model circuits incorporating qubit decoherence as an error factor. Typically, coherence is specified as an interval corresponding to a worst and a best case. We employ the two accessibility relations in a PLTS to model both scenarios simultaneously.

An important observation for the conversion of quantum circuits to PLTS is that quantum circuits always have a sequential execution. Simultaneous operations performed to distinct qubits are combined using the tensor product  $\otimes$  into a single operation to the whole collection of qubits which forms the state of the circuit. The latter is described by a sequence of executions  $e_1, e_2, e_3, \dots$  where each  $e_i$  is the tensor product of the operations performed upon the state at each step. The conversion to a PLTS is straightforward, labelling each transition by the tensor of the relevant gates  $O_1 \otimes \dots \otimes O_m$ , for  $m$  gates involved, but for the computation of the positive and negative accessibility relations,  $r^+$  and  $r^-$ .

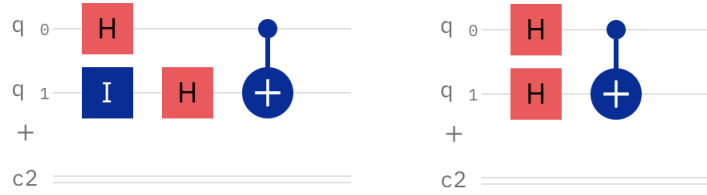
The weights of a transition corresponding to the application of a gate  $O$  acting over  $n$  qubits  $q_1$  to  $q_n$  are given by

$$v(O) = \begin{cases} (1, 0) & \text{if qubits } q_1, \dots, q_n \text{ are in a definite state} \\ (\text{Max}_i f_{\max}(q_i), \text{Min}_i f_{\min}(q_i)) & \text{otherwise} \end{cases}$$

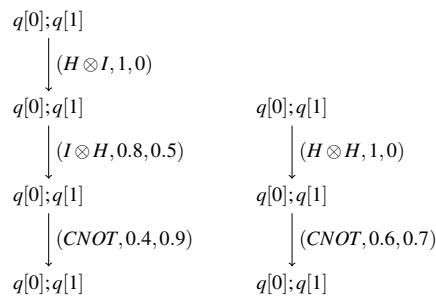
where  $f_{\max}(q) = \frac{\tau_{\max}(q) - \tau_{\text{prep}}(q)}{100}$  and  $f_{\min}(q) = \frac{\tau_{\min}(q) - \tau_{\text{prep}}(q)}{100}$ ,  $\tau_{\max}(q)$  and  $\tau_{\min}(q)$  are the longest and shortest coherence times of  $q$ , respectively, and  $\tau_{\text{prep}}(q)$  is the time from the preparation of  $q$ 's superposition to the point after the execution of  $O$ . The latter are fixed for each type of quantum gate; reference [14] gives experimentally computed values for them as well as for maximum and minimum values for qubit decoherence.

Consider, now, a transition  $t$  labelled by a  $O_1 \otimes \dots \otimes O_m$ . Then,  $r^+ = \text{Max}_{i=1}^n \{\pi_1(v(O_i))\}$  and  $r^- = 1 - \text{Min}_{i=1}^n \{\pi_2(v(O_i))\}$ .

**Example 10.** Consider the following circuits designed with IBM Quantum Composer:



Assume that the execution time of a single qubit gate is  $\tau_G = 20\mu s$  and of a two qubit gate is  $2\tau_G = 40\mu s$  [14], and that both qubits have the same coherence times  $\tau_{max}(q_1) = \tau_{max}(q_2) = 100\mu s$  and  $\tau_{min}(q_1) = \tau_{min}(q_2) = 70\mu s$ . Thus the circuit on the left (resp. right) translates into  $T_1$  (on the left) and  $T_2$  (on the right).



As both circuits implement the same quantum algorithm and our focus is only on the effectiveness of the circuits, we may abstract from the actual sequences of labels and consider instead  $T_1\{\lambda\}$  and  $T_2\{\lambda\}$ , for  $\lambda$  mapping each label to a unique label  $\star$ . Their maximal weighted traces <sup>2</sup> are

$$t_{T_1\{\lambda\}} = \langle [* , * , *], 0.4, 0.9 \rangle \text{ and } t_{T_2\{\lambda\}} = \langle [* , * , *], 0.6, 0.7 \rangle$$

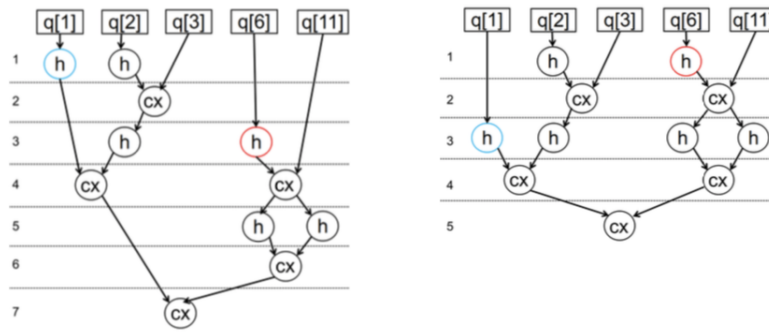
Clearly  $t_{T_1\{\lambda\}}$  is a weighted subtrace of  $t_{T_2\{\lambda\}}$ , therefore suggesting a criteria for comparing the effectiveness of circuits. Indeed, a circuit is more effective (i.e. less affected by qubit decoherence) than other if the maximal weighted trace of its (relabelled) PLTS is a weighted subtrace of the corresponding construction in the other.

The second circuit is obviously more efficient than the first. This suggests we could use the weighted subtrace relation as a metric to compare circuit quality, for circuits implementing equivalent algorithms.

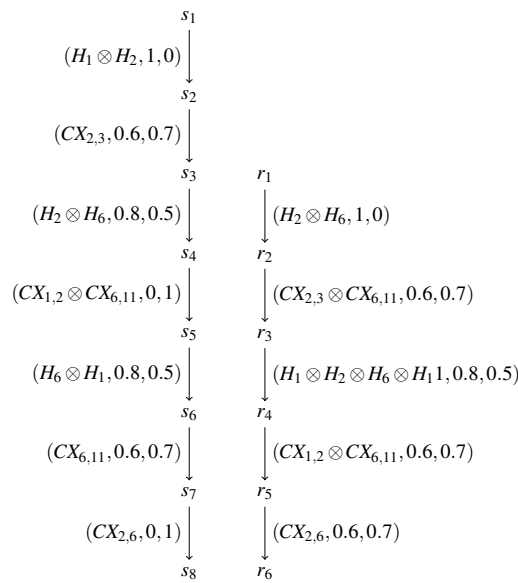
Reference [14] introduces a tool which tried to transform a circuit so that the lifetime of quantum superpositions is shortened. They give several examples of circuits and show how the application of the tool results in a circuit performing the same algorithm but with a reduced error rate. Our next example builds on one of their examples, computes the corresponding PLTS and compare the maximal weighted traces.

**Example 11.** Consider the following circuits reproduced from [14], which in ideal quantum devices would be indistinguishable.

<sup>2</sup>Such maximal traces are easily identifiable given the peculiar shape of a PLTS corresponding to a quantum circuit.



These circuits are represented as

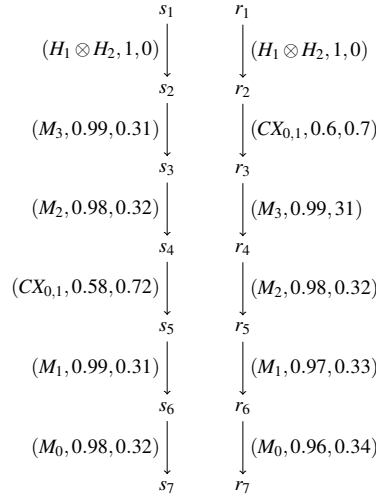


where  $H$  and  $CX$  are indexed by the numeric identifiers of the qubit(s) to which they apply in each execution step. The maximal weighted trace of the (relabelled PLTS corresponding to) circuit in the right,  $\langle [*, *, *, *, *, *, *], 0.6, 0.7 \rangle$ , is a weighted subtrace of the one corresponding to circuit in the left,  $\langle [*, *, *, *, *, *], 0, 1 \rangle$ . Thus, the former circuit is more effective than the latter, as experimentally verified in [14].

**Example 12.** As a final example consider two circuits differing only on the time points in which measurements are placed.



The corresponding PLTS, computed again with the values given in reference (where execution time of a measurement is  $\tau_M = 300ns \sim 1\mu s$ ), are depicted below



The maximal weighted trace  $\langle [*, *, *, *, *, *], 0.6, 0.7 \rangle$  corresponding to the circuit on the right is a weighted subtrace of the corresponding one for the circuit on the left,  $\langle [*, *, *, *, *, *], 0.58, 0.72 \rangle$ . This shows that measuring can be safely postponed to the end of a circuit, as experimentally verified.

## 6 Conclusions and future work

The paper introduced a category of a new kind of labelled transition systems able to capture both *vagueness* and *inconsistency* in software modelling scenarios. The structure of this category was explored to define a number of useful operators to build such systems in a compositional way. Finally, PLTS were used to model effectiveness concerns in the analysis of quantum circuits. In this case the weight corresponding to the ‘presence’ of a transition captures an index measuring its effectiveness assuming the best case value for qubit decoherence. On the other hand, the weight corresponding to the ‘absence’ of a transition measures the possibility of non-occurrence, assuming qubit decoherence worst case value.

A lot remains to be done. First of all, a process logic, as classically associated to labelled transition systems [12], i.e. a modal logic with label-indexed modalities, can be designed for pointed PLTS. This will provide not only yet another behavioural equivalence, based on the set of formulas satisfied by two systems, but also a formal way to express safety and liveness properties of these systems.

This will be extremely useful to express and verify properties related to the effectiveness of quantum circuits, therefore pushing further the application scenario proposed in section 5. Finally, automating the construction of a pointed PLTS for a given circuit, parametric on the different qubit coherence and gate execution time found experimentally, and adding a prover for the logic suggested above, will provide an interesting basis to support quantum circuit optimization. Reliable, mathematically sound approaches and tools to support quantum computer programming and verification will be part of the quantum research agenda for the years to come. Indeed, their lack may put at risk the expected quantum advantage of the new hardware.

## References

- [1] Agudelo, J.C.A., Carnielli, W.A.: Paraconsistent machines and their relation to quantum computing. *J. Log. Comput.* **20**(2), 573–595 (2010), <https://doi.org/10.48550/arXiv.0802.0150>
- [2] Akama, S. (ed.): *Towards Paraconsistent Engineering*, Intelligent Systems Reference Library, vol. 110. Springer (2016), <https://doi.org/10.1007/978-3-319-40418-9>
- [3] Baeten, J.C.M., Basten, T., Reniers, M.A.: *Process Algebra: Equational theories of communicating processes*. Cambridge Tracts in Theoretical Computer Science (50), Cambridge University Press (2010), <https://doi.org/10.1017/CB09781139195003>
- [4] Chiara, M.L.D., Giuntini, R.: Paraconsistent ideas in quantum logic. *Synth.* **125**(1-2), 55–68 (2000), <https://doi.org/10.1023/A:1005296018904>
- [5] da Costa, N.C.A., Krause, D., Bueno, O.: Paraconsistent logics and paraconsistency. In: Jacquette, D. (ed.) *Handbook of the Philosophy of Science (Philosophy of Logic)*. pp. 791–911. Elsevier (2007), <https://doi.org/10.1016/B978-044451541-4/50023-3>
- [6] da Costa, N.C.A., Krause, D.: Physics, inconsistency, and quasi-truth. *Synth.* **191**(13), 3041–3055 (2014), <https://doi.org/10.1007/s11229-014-0472-8>
- [7] Cruz, A., Madeira, A., Barbosa, L.S.: A logic for paraconsistent transition systems. In: Indrzejczak, A., Zawidzki, M. (eds.) *Proceedings of the 10th International Conference on Non-Classical Logics. Theory and Applications, NCL 2022, Łódź, Poland, 14-18 March 2022. EPTCS*, vol. 358, pp. 270–284 (2022), <https://doi.org/10.4204/EPTCS.358.20>
- [8] Cruz, A.L.R.: *Exploring paraconsistent logics for quantum programs*. MSc Thesis in Engineering Physics, DI, Universidade do Minho (2021)
- [9] Jaśkowski, S.: Propositional calculus for contradictory deductive systems. *Studia Logica* **24**(1), 143–157 (1969), <https://doi.org/10.1007/BF02134311>
- [10] Nielsen, M.A., Chuang, I.L.: *Quantum Computation and Quantum Information (10th Anniversary Edition)*. Cambridge University Press (2010), <https://doi.org/10.1017/CB09780511976667>
- [11] Preskill, J.: Quantum computing in the nisq era and beyond. *Quantum* **2**(79), 87–95 (2018), <https://doi.org/10.22331/q-2018-08-06-79>
- [12] Stirling, C.: *Modal and Temporal Properties of Processes*. Texts in Computer Science, Springer (2001), <https://doi.org/10.1007/978-1-4757-3550-5>
- [13] Winskel, G., Nielsen, M.: Models for concurrency. In: Abramsky, S., Gabbay, D.M., Maibaum, T.S.E. (eds.) *Handbook of Logic in Computer Science (vol. 4): Semantic Modelling*, pp. 1–148. Oxford Science Publications (1995)
- [14] Zhang, Y., Deng, H., Li, Q., Song, H., Nie, L.: Optimizing quantum programs against decoherence: Delaying qubits into quantum superposition. In: *2019 Int. Symp. Theoretical Aspects of Software Engineering (TASE)*. IEEE (Jul 2019), <https://doi.org/10.48550/arXiv.1904.09041>