Acta Cybernetica **23** (2017) 327–347.

On the Completeness of the Traced Monoidal Category Axioms in (Rel,+)[*]

Miklós Bartha[a]

To the memory of my friend and former colleague Zoltán Ésik

Abstract

It is shown that the traced monoidal category of finite sets and relations with coproduct as tensor is complete for the extension of the traced symmetric monoidal axioms by two simple axioms, which capture the additive nature of trace in this category. The result is derived from a theorem saying that already the structure of finite partial injections as a traced monoidal category is complete for the given axioms. In practical terms this means that if two biaccessible flowchart schemes are not isomorphic, then there exists an interpretation of the schemes by partial injections which distinguishes them.

Keywords: monoidal categories, trace, iteration, feedback, identities in categories, equational completeness

1 Introduction

It was in March, 2012 that Zoltán visited me at the Memorial University in St. John's, Newfoundland. I thought we would find out a research topic together, but he arrived with a specific problem in mind. He knew about the result found by Hasegawa, Hofmann, and Plotkin [12] a few years earlier on the completeness of the category of finite dimensional vector spaces for the traced monoidal category axioms, and wanted to see if there is a similar completeness statement true for the matrix iteration theory of finite sets and relations as a traced monoidal category. Unfortunately, during the short time we spent together, we could not find the solution, and after he had left I started to work on some other problems. Having learned about his shocking untimely death, I felt impelled to finally solve the problem that we started to work on together, and have it published in his memory.

The result obtained in this paper points beyond the original goal of finding a suitable extension of the traced monoidal category axioms for which the category

[*]Work initiated by Zoltán Ésik as part of a research project in 2012.

[a]Department of Computer Science, Memorial University of Newfoundland, St. John's, NL, Canada, E-mail: bartha@mun.ca

DOI: 10.14232/actacyb.23.1.2017.18

$(\mathbf{Rel}_{fin}, +)$ is complete. We show that already the category $(\mathbf{Pin}_{fin}, +)$ of finite partial injections is complete for the minimal extension of the traced monoidal axioms that can be expected from these structures. Since $(\mathbf{Pin}_{fin}, +)$ is a subcategory of $(\mathbf{Rel}_{fin}, +)$, the completeness result originally sought by Zoltán follows as a corollary.

We shall assume familiarity with the basic concepts of category theory [19], and it helps if the reader is also familiar with Zoltán's work on iteration theories [9].

2 Traced monoidal categories

A symmetric *monoidal category* consists of a category $\mathcal{C}$ equipped with a bifunctor $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ called *tensor*, and a unit object I of $\mathcal{C}$. Furthermore, $\mathcal{C}$ has natural isomorphisms:

$$a_{X,Y,Z} : (X \otimes Y) \otimes Z \to X \otimes (Y \otimes Z)$$

called *associators*,

$$l_X : I \otimes X \to X \text{ and } r_X : X \otimes I \to X$$

called left and right *unitors*, and

$$c_{U,V} : U \otimes V \to V \otimes U$$

called *symmetries*, which isomorphisms are subject to the standard coherence axioms given in [19]. The monoidal category $\mathcal{C}$ is *strict* if the bifunctor $\otimes$ is strictly associative, so that

$$A \otimes (B \otimes C) = (A \otimes B) \otimes C \text{ and } A \otimes I = I \otimes A = A,$$

and all of the associators and unitors are the identities.

Let $\mathcal{C}$ be a monoidal category with tensor $\otimes$ and unit object I as specified above. The following definition of traced monoidal categories uses the terminology of [15], except that trace in is introduced as *left* trace, that is, an operation $\mathcal{C}(U + A, U + B) \to \mathcal{C}(A, B)$, rather than $\mathcal{C}(A + U, B + U) \to \mathcal{C}(A, B)$ (i.e., right trace) as it appears in [15]. The reason is to remain consistent with the notation used in Zoltán's and the author's own work. Also, following in Zoltán's footsteps, we write composition of morphisms in a left-to-right way, that is, the composite of $f : A \to B$ and $g : B \to C$ is $f \circ g : A \to C$.

Definition 1. A *trace* for $\mathcal{C}$ is a family of functions

$$Tr^U_{A,B} : \mathcal{C}(U \otimes A, U \otimes B) \to \mathcal{C}(A, B)$$

natural in A and B, satisfying the following four axioms:

vanishing:

$$Tr^{U\otimes V}_{A,B}(a_{U,V,A} \circ f) = Tr^V_{A,B}(Tr^U_{V\otimes A,V\otimes B}(f \circ a_{U,V,B})),$$

where $f : U \otimes (V \otimes A) \to (U \otimes V) \otimes B$;

superposing:

$$Tr^U_{A,B}(f) \otimes g = Tr^U_{A\otimes C,B\otimes D}(a^{-1}_{U,A,C} \circ (f \otimes g) \circ a_{U,B,D}),$$

where $f : U \otimes A \to U \otimes B$, and $g : C \to D$;

sliding:

$$Tr^U_{A,B}((g \otimes 1_A) \circ f) = Tr^V_{A,B}(f \circ (g \otimes 1_B))$$

where $f : V \otimes A \to U \otimes B$ and $g : U \to V$;

yanking:

$$Tr^U_{U,U}(c_{U,U}) = 1_U.$$

It is well-known that sliding of symmetries (whereby g is chosen as a symmetry isomorphism in the sliding axiom) suffices in the presence of the other axioms. Also, the special vanishing axiom:

$$Tr^I_{A,B}(l_A \circ f \circ l^{-1}_B) = f \quad \text{for } f : A \to B,$$

which was part of the original system in [15], can easily be derived from the other axioms and can therefore be omitted. As the reader can see, the necessity of using the associator and unitor morphisms $a(X, Y, Z)$, l_X in the traced monoidal axioms makes them look very complicated, even though their graphical representation below shows that in fact they are quite intuitive and simple. It is known, see e.g. [20], that every monoidal category is equivalent to a strict one. Quoting an argument from [14], "most results obtained with the hypothesis that a monoidal category is strict can, in principle, be reformulated and proved without that condition". Our result in this paper is no exception, therefore in the sequel we shall make the technically simplifying assumption that our monoidal categories (traced or not) are strict.

The graphical representation of the traced monoidal category axioms (with the strictness assumption incorporated) is given in Figures 1–5.

Traced monoidal categories (with one additional axiom) and their graphical language first appeared in [1] in an algebraic setting, using the name "scheme algebra" for these structures. The operation corresponding to trace was called feedback. The year was 1987, and already at that time Zoltán and Steve Bloom had a significant number of important results on iterative and iteration theories, the study of which was initiated by Calvin C. Elgot in the early 1970s. The motivation of that study was to find out the equational laws characterizing the iteration operation in flowchart-related algorithms. The algebra (category) of flowcharts itself has also been axiomatized in terms of the iteration operation [8]. This axiomatization was a little awkward, however, because the iteration operation

$$f^\dagger : n \to p \quad \text{for } f : n \to n + p,$$

where n and p are non-negative integers, was intended to capture the semantical aspects of iteration in the first place. For syntactical purposes the feedback

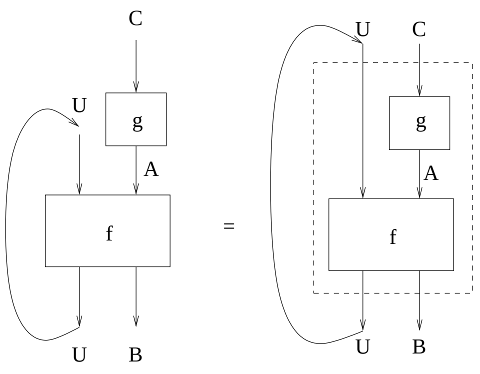

Figure 1: Naturality of trace in A.

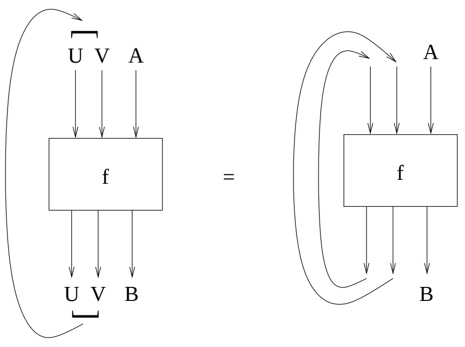

Figure 2: Vanishing

operation

$$\uparrow^n f : p \to q \quad \text{for } f : n + p \to n + q$$

turns out to be a lot more practical and easier to deal with. No loss of generality arises from the switch, provided that the underlying structures are algebraic theories, since there are standard rewriting rules between iteration and feedback in all such theories. See [9, 1, 25]. (The purely syntactical category of flowcharts is of course not such a structure.) Regarding the exact relationship between traced monoidal categories and iteration theories, it was proved in [1] that a single-sorted traced monoidal category is an iteration theory iff it is an algebraic theory and satisfies the commutative axioms discovered by Zoltán in [11].

Continuing the story of traced monoidal categories, essentially the same axiomatization and graphical language as the one presented in [1] was published a few years later in [10] under the name "biflow". Neither of these pioneer works noticed, however, that the monoidal category of finite dimensional vector spaces, in which tensor is tensor product and feedback is trace, is an obvious scheme algebra/biflow, provided that the axiomatization is lifted from the single-sorted algebraic language

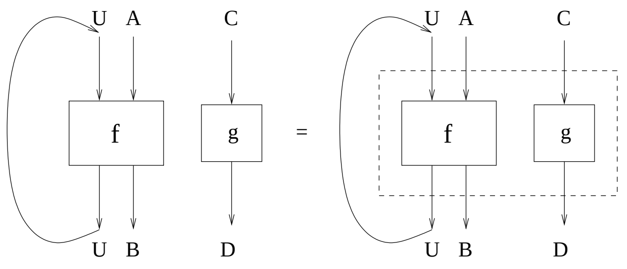

Figure 3: Superposing

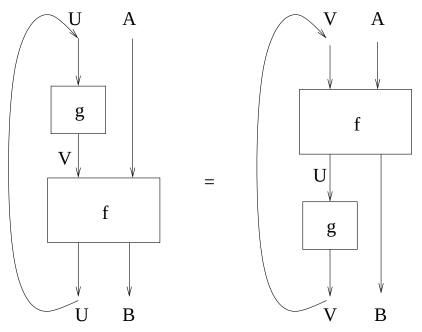

Figure 4: Sliding

to the general "polymorphic" categorical one. Finally, in 1996, Joyal, Street, and Verity made this important point by essentially rediscovering the old scheme axioms in a general new context, which also covered balanced monoidal categories. They also presented the fundamental *Int* construction on the embedding of an arbitrary balanced traced monoidal category into a tortile one [15]. In case braiding is symmetry, as it is in our present study, the *Int* construction transforms an arbitrary traced monoidal category into a compact closed category [17].

By virtue of the above discussion, the feedback operation is deeply rooted in control theory, whereas trace is a concept used primarily in finite dimensional vector spaces as an operation on linear maps (matrices). The usual interpretations of trace and feedback have not much in common, since trace is "multiplicative" style in contrast with feedback, which has a strong "additive" flavour. The informal distinction "additive or multiplicative" uses the very basic category of sets and functions as a basis for comparison. Taking tensor in this category as Cartesian product with $I = \{\emptyset\}$ (multiplicative) or coproduct with $I = \emptyset$ (additive) results in entirely different monoidal categories. On this basis, "multiplicative" in vector spaces means that $\otimes$ is tensor product rather than ordinary product, which happens to coincide with coproduct.

According to the main result of [1], the free single-sorted category with feedback generated by a collection of morphisms (boxes) is the algebra of flowchart schemes,

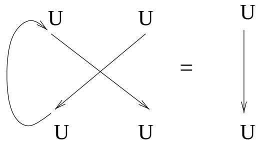

Figure 5: Yanking

which is definitely additive style and reflects the flow of information in flowchart algorithms. In summary, there are countless reasons to call the operation

$$Tr^U_{A,B} : \mathcal{C}(U \otimes A, U \otimes B) \to \mathcal{C}(A, B)$$

feedback, and just a few to call it trace. Nevertheless, the name "trace" stuck, and today everyone calls the categorical structures corresponding to scheme algebras traced monoidal categories.

To complicate the issue even further, categories with feedback have also been considered in [16] and a series of works by the present author [2, 3, 4, 5]. In these categories, however, yanking is missing from the axioms imposed. Feedback in such categories is *delayed* like in synchronous systems, e.g. sequential circuits. The meaning of the loop on the left-hand-side of the yanking axiom (Fig. 5) is a *register*, a memory element, which suggests a step-by-step behavior for synchronous systems (circuits/schemes). Note that this kind of categorical interpretation over sets as objects is multiplicative style, since a morphism $A \to B$ is a Mealy automaton $U \times A \to U \times B$ with U being an arbitrary set (of states). As a consequence, there is no explicit control present in the system. In a sequential circuit, for example, every logical gate and flip-flop is engaged in each clock cycle, so that the whole system is massively parallel. What we call "control" in the von Neumann computer architecture is just an abstraction, an extra control line carrying a digital information indicating that the control is present or not. Nevertheless, if the underlying monoidal category of the category of circuits (automata) is additive, then the automata themselves will be additive, too, possessing the equivalent of some kind of control.

As stated in the Introduction of [9], and demonstrated throughout the book, iteration theories are ubiquitous in computer science. If this is true, then traced monoidal categories are twice as ubiquitous and not only in computer science but in the whole of mathematics. Indeed, these categories are more general than iteration theories, therefore they cover more ground. Here are just a few among the most important traced monoidal categories occurring with great frequency.

1. *The additive category* $(\mathbf{Rel}, +)$ *of relations with* $\otimes$ *being coproduct.*

The base category of $(\mathbf{Rel}, +)$ is the category $\mathbf{Rel}$ of sets and relations. That is, objects are sets, and a morphism $A \to B$ is a relation $R \subseteq A \times B$. Tensor of objects is disjoint union $(+)$, $I = \emptyset$, and tensor of morphisms is disjoint union of relations.

For $R : U + A \to U + B$, $Tr^U_{A,B}R$ is given by

$$(a, b) \in Tr^U_{A,B}R \text{ iff } \exists\, u_1, \ldots, u_n \in U,\ n \geq 0, \text{ such that } aRu_1R \ldots Ru_nRb.$$

The category $(\mathbf{Rel}, +)$ restricts naturally to $\mathbf{Rel}_{fin}$, the category of finite sets and relations. This restriction is equivalent to its full subcategory $(\mathbf{Rel}_{\mathbb{N}}, +)$ induced by the objects in $\mathbb{N} = \{0, 1, \ldots n \ldots\}$, where $0 = \emptyset$ and $n + 1 = n \cup \{n\}$ as in Zermelo-Fraenkel set theory. This subcategory is strict and it is closed for trace. It is also single-sorted, since the set of objects $\mathbb{N}$ is generated by the object 1 using tensor. Clearly, $I = 0$ in $(\mathbf{Rel}_{\mathbb{N}}, +)$.

It is very instructive to look at the matrix representation of $(\mathbf{Rel}_{\mathbb{N}}, +)$, for this category is a matrix iteration theory as well. It was shown in [9, Corollary 5.5] that this theory is generated by the initial ω-idempotent Boolean semiring $\mathbf{B}$. In other words, if relations are represented as 0-1 matrices, then composition and tensor (coproduct!) of relations is that of matrices, using the algebraic rules of $\mathbf{B}$ rather than those of GF(2). (That is, 1+1=1 and not 0.) The trace of a finite relation $R : n + p \to n + q$ having a matrix decomposition

$$R = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

with $A : n \to n$, $B : n \to q$, $C : p \to n$, $D : p \to q$ can then be obtained by the well-known Kleene formula:

$$Tr^n_{p,q}R = D + CA^*B, \tag{1}$$

where A^* denotes the infinite sum

$$A^* = \sum_{k=0}^{\infty} A^k$$

according to the Boolean semiring addition and multiplication rules. That is, in the Kleene formula (1), $+$ denotes the Boolean sum of matrices $p \to q$ (rather than $+$ as tensor/coproduct). The present definition of the Kleene star * coincides with the star operation used in matrix iteration theories.

2. *The multiplicative category* $(\mathbf{Rel}, \times)$ *of relations with product as tensor.*

The base category of $(\mathbf{Rel}, \times)$ is also $\mathbf{Rel}$, but tensor is $\times$ (Cartesian product of sets) rather than $+$. The object I is $\{\emptyset\}$. The tensor of two relations is the product of them in the usual sense. For $R : U \times A \to U \times B$, its trace is defined by

$$(a, b) \in Tr^U_{A,B}R \text{ iff } \exists u \in U \text{ such that } ((u, a), (u, b)) \in R.$$

Again, $(\mathbf{Rel}, \times)$ restricts to the category $\mathbf{Rel}_{fin}$ of finite sets and relations, which category is equivalent to its full subcategory $(\mathbf{Rel}_{\mathbb{N}}, \cdot)$ induced by the objects $\mathbb{N}$, provided that the "set" $n \times m$ is identified with $n \cdot m$ in a given canonical way (e.g. enumeration by rows or columns). In everyday language, take the matrix representation of relations. The unit object I is 1. The category $(\mathbf{Rel}_{\mathbb{N}}, \cdot)$ is no longer single-sorted, but it is still generated by the prime numbers (and 0) as objects, and it is strict.

3. *The multiplicative category* $(\mathbf{FdVect}_K, \otimes)$ *of finite dimensional vector spaces over a given field* K.

The base category is $\mathbf{FdVect}_K$, and $\otimes$ is tensor product of linear maps. The object I is the field K as a 1-dimensional vector space. Analogously to $(\mathbf{Rel}, \times)$, this category is also equivalent to its restriction induced by the concrete n-dimensional spaces F_K^n, in which linear maps are simply $n \times m$ matrices. In this context we can even identify the object F_K^n with the number n, since K is fixed. The reduced category is again strict. For a linear map (matrix, for simplicity) $M : U \to U$, $Tr^U_{I,I} M$ is the sum of diagonal elements in M. The general definition of trace is technically more complicated, and the reader is referred to [15] for the details of this definition. The analogy between $(\mathbf{Rel}_{fin}, \times)$ and $(\mathbf{FdVect}_K, \otimes)$ is that, when relations are represented as 0-1 matrices, their tensor and trace is calculated in the exact same way as in (the strict equivalent of) $(\mathbf{FdVect}_K, \otimes)$, interpreting the ring operations according to the Boolean semiring $\mathbf{B}$ rather than the field K. Since $\mathbf{B}$ is just a commutative semiring (addition is idempotent in $\mathbf{B}$), the morphisms of $(\mathbf{Rel}_{fin}, \times)$ are not linear maps between vector spaces. (They are just linear maps of *free semimodules* over the semiring $\mathbf{B}$.)

4. *The additive category* $(\mathbf{Iso}, \oplus)$ *of quantum control.*

Let $\mathbf{FdHilb}$ denote the category of finite dimensional Hilbert spaces. The base category $\mathbf{Iso}$ is then the subcategory of $\mathbf{FdHilb}$ having isometries only as its morphisms. Tensor is now $\oplus$, that is, coproduct/product in $\mathbf{FdHilb}$. The unit object I is the zero space. Notice the drastic change from the multiplicative tensor $\otimes$ in $(\mathbf{FdVect}_K, \otimes)$ to the additive $\oplus$, which is analogous to $+$ in $(\mathbf{Rel}_{fin}, +)$ as a matrix theory.

Let $\tau : \mathcal{U} \oplus \mathcal{K} \to \mathcal{U} \oplus \mathcal{L}$ be an isometry. Then $Tr^{\mathcal{U}}_{\mathcal{K},\mathcal{L}} \tau : \mathcal{K} \to \mathcal{L}$ is the isometry specified as follows. Consider the matrix of τ

$$\begin{array}{cc} & \begin{array}{cc} \mathcal{U} & \mathcal{L} \end{array} \\ \begin{array}{c} \mathcal{U} \\ \mathcal{K} \end{array} & \left(\begin{array}{cc} \tau_A & \tau_B \\ \tau_C & \tau_D \end{array} \right) \end{array}$$

according to the biproduct decomposition

$$\tau = \langle [\tau_A, \tau_C], [\tau_B, \tau_D] \rangle,$$

where $[\_]$ stands for coproduct and $\langle \_ \rangle$ for product. Trace is defined again by the Kleene formula

$$Tr^{\mathcal{U}}_{\mathcal{K},\mathcal{L}} \tau = \lim_{n \to \infty} (\tau_D + \tau_C \circ \tau_A^{*n} \circ \tau_B). \qquad (2)$$

In the present Kleene formula

$$\tau_A^{*n} = \sum_{i=0}^{n} \tau_A^i,$$

where $\tau_A^0 = I_{\mathcal{U}}$ and $\tau_A^{i+1} = \tau_A^i \circ \tau_A$. In other words, τ_A^{*n} is the n-th approximation of τ_A's *Neumann series* well-known in operator theory.

It was proved in [7] that the limit in (2) always exists and the convergence is strong, resulting in an isometry. This result is very surprising, for the monoidal category $(\mathbf{Iso}, \oplus)$ does not even resemble to an algebraic theory and yet, it has a trace completely analogous to iteration in matrix iteration theories. Also, the ring operations addition and multiplication performed in matrix composition and sum are over the *field* $\mathbb{C}$, not over the trivial Boolean *semiring* $\mathbf{B}$. The Kleene formula (2) does not work in the whole category $(\mathbf{FdHilb}, \oplus)$ [21], and there appears to be no way to find a reasonable trace for this monoidal category.

The additive style quantum trace explains how the flow of quantum information can be controlled in a quantum flowchart algorithm [23] or a Turing machine [7]. At the moment its consistency is only proved for finite dimensional Hilbert spaces, which is insufficient to explain the semantics of general Turing machines with an infinite number of tape cells (or the semantics of a von Neumann style analog quantum computer architecture having an infinite memory component). Further analysis is needed to generalize the quantum trace for separable (i.e. countably infinite dimensional) Hilbert spaces. This issue is of utmost importance in the logical design of quantum computers.

At this point the reader might wonder why the very basic monoidal category $(\mathbf{Set}, \times)$ is missing from the above list of examples. What is a reasonable trace for this category, or a suitable extension of it? At the moment this question is still unanswered. Trace cannot simply be adopted from $(\mathbf{Rel}, \times)$, because the trace of a function might turn out to be a relation only. Thus, a deterministic computation would trigger a nondeterministic one, which we certainly do not want to allow. The author proved in [4] that the category with (delayed) circular feedback freely generated from $(\mathbf{Set}_{fin}, \times)$ is the category $Sim(\mathbf{Set}_{fin}, \times)$ of simulation equivalent finite state deterministic Mealy automata. No delay-free model is currently known.

3 Looking at trace in the Hungarian way

In this section we relate the trace operation in $(\mathbf{Rel}_{fin}, +)$ to bipartite graph matchings, and show how the trace of a relation $R : U + A \to U + B$ can be constructed by the help of the well-known Hungarian method. We also take a closer look at Selinger's construction [22] of embedding the category $(\mathbf{Rel}, +)$ into $(\mathbf{Rel}, \times)$, and interpret this embedding as a network flow problem. Let $R : U + A \to U + B$ be a relation over finite sets. The standard graph representation of R is a bipartite graph $G_R = (U + A, U + B)$. See Fig. 6. Extend G_R by edges connecting each vertex $u \in U$ in bipartition $U + A$ with the corresponding u in $U + B$. See Fig. 7. Let G_R^U denote the resulting graph. Consider the collection of newly added edges as a matching M in G_R^U, and construct the relation $R_M : A \to B$ in the following way. For each $(a, b) \in A \times B$, put (a, b) in R_M iff there is an augmenting path in G_R^U from a to b with respect to M. Recall from [18] that an augmenting path is an M-alternating path starting and ending at vertices not covered by M. It is

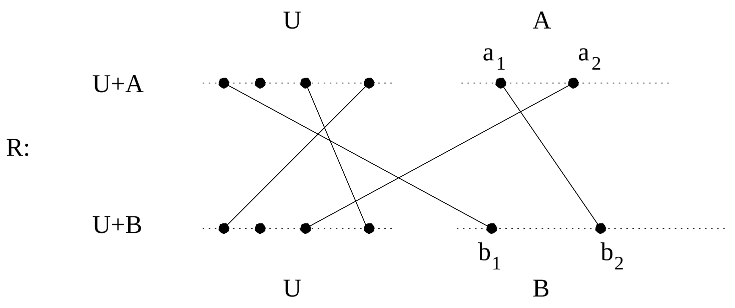

Figure 6: The graph of relation R

immediate by the definitions that

$$(a,b) \in Tr^{U}_{A,B}R \;\text{ iff }\; (a,b) \in R_M.$$

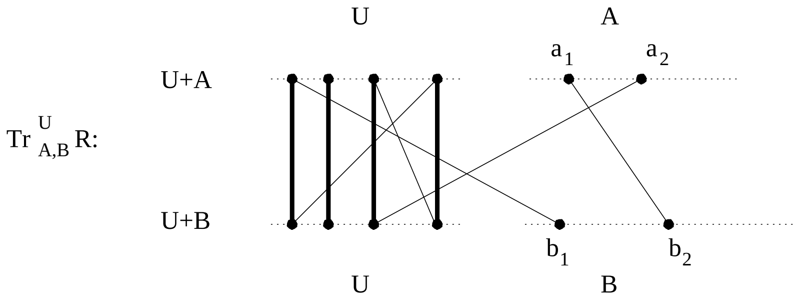

Figure 7: The graph G^U_R with matching M

Thus, in order to calculate $Tr^U_{A,B}R$, it is sufficient to apply the Hungarian method on the graph G^U_R with matching M to find a maximum matching in G^U_R. This is done by looking at the Hungarian forest F obtained at the final stage of the algorithm and see which vertex pairs (a,b) are in the same tree of F. See [18] for the details of the Hungarian method.

The functorial embedding of $(\mathbf{Rel},+)$ into $(\mathbf{Rel},\times)$ by Selinger [22] is yet another example of traced monoidal categories emerging naturally in combinatorial optimization problems. Let $R : A \to B$ be a relation between finite sets A and B, and consider the bipartite graph $G_R = (A,B)$ corresponding to R. Interpret G_R as a network flow problem [18] by assigning non-negative integers to the vertices in A and B. The assignment $f_A : A \to \mathbb{N}$ specifies the *supply* of some merchandise available at each vertex (warehouse) $a \in A$, while the assignment $f_B : B \to \mathbb{N}$ captures the *demand* on the B side for the same merchandise. Assume that

$$t_f = \sum_{a\in A} f_A(a) = \sum_{b\in B} f_B(b)$$

holds, so that the total supply meets the total demand.

The task is to deliver all the goods from side A to side B (in one round) along the roads (edges) between the two sides according to G_R. See Fig. 8. Clearly, a

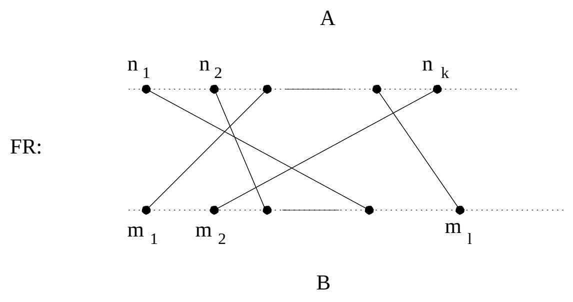

Figure 8: The network flow problem for R

solution to the problem is an assignment $\rho : E(G_R) \to \mathbb{N}$ satisfying the conditions

$$f_A(a) = \sum_{b' \in B} \rho(a, b') \quad \text{and} \quad \sum_{a' \in A} \rho(a', b) = f_B(b)$$

for all $a \in A$ and $b \in B$.

The functor $F : (\mathbf{Rel}, +) \to (\mathbf{Rel}, \times)$ is now defined as follows.

On objects: For every set A, $FA = [A \to \mathbb{N}]_{fin}$, where $[A \to \mathbb{N}]_{fin}$ denotes the set of functions $f : A \to \mathbb{N}$ by which $f(a) = 0$ for all but finitely many $a \in A$.

On morphisms: If $R : A \to B$ is a relation, then for every $f_A \in FA$ and $f_B \in FB$,

$$(f_A, f_B) \in FR \ \text{ iff the network flow problem } (f_A, f_B) \text{ in } G_R \text{ has a solution.}$$

Note that the concrete problem (f_A, f_B) is finite, because both f_A and f_B are in $[A \to \mathbb{N}]_{fin}$.

It was proved in [22] that F is a functorial embedding of traced monoidal categories. Selinger has also proved that there exists no such embedding of $(\mathbf{Rel}_{fin}, +)$ into $(\mathbf{Rel}_{fin}, \times)$. In particular, for the restriction of our concrete embedding F to $(\mathbf{Rel}_{fin}, +)$, one cannot assume that the supply and demand numbers assigned by f_A and f_B to the vertices of G_R remain under a fixed upper bound.

4 Free traced monoidal categories

In general, a *coherence result* for a subcategory μ of monoidal categories is about establishing a left adjoint for a forgetful functor F from the category of μ-monoidal categories into an appropriate syntactical category, and providing a graphical characterization of the free monoidal μ-categories so obtained. For some typical examples, see [19, 20, 17, 24, 1, 2]. In this section we briefly overview the construction of the free traced monoidal category generated by a set of morphisms (variables) presented in [6]. We shall maintain the assumption that the monoidal category to be constructed is strict, even though it is very simple to modify the construction to obtain the non-strict free traced monoidal categories. Our way of choosing the forgetful functor F differs from the method followed e.g. in [17], where the category structure was still preserved. We go one step further in forgetting, and preserve only the alphabet structure of morphisms. To remain consistent with set theory, assume that the hom-sets of our monoidal categories are indeed sets.

For a class O of object variables, a *doubly ranked alphabet* or *monoidal signature*

$$\Sigma = (O, M, r)$$

[14] consists of a set M of *morphism variables* and a mapping r which assigns for each variable $f \in M$ a *domain* $dom(f)$ and a *codomain* $cod(f)$, which are finite sequences (strings) over O. The pair $(dom(f), cod(f))$ is called the *rank* of f. As it is natural, we write $f : dom(f) \to cod(f)$. In case O is the set $\mathbb{N}$, the standard concept of doubly ranked alphabet is recaptured. An *alphabet mapping* between ranked alphabets $\Sigma = (O, M, r)$ and $\Delta = (O', M', r')$ is a mapping ϕ, which assigns to each object variable A in O an object variable ϕA in O' and to each morphism variable $f : u \to v$ in M a morphism variable $\phi f : \phi u \to \phi v$ in M', where ϕ is extended to strings in O^* in the natural way. Thus, an alphabet mapping preserves the rank of morphism variables.

Every (strict) monoidal category $\mathcal{M}$ can trivially be considered as a ranked alphabet $\Sigma = \mathcal{AM}$ in which the object variables are the objects of $\mathcal{M}$ (denoted by O_M as a concrete monoid structure). As an important twist, however, the empty string ϵ in O_M^* must be identified with the object I in O_M. The morphism variables of Σ with rank $u \to v$ are simply all the morphisms $|u| \to |v|$ in $\mathcal{M}$, where $|u|$ and $|v|$ are the evaluations of u and v in the given monoid structure on O_M. We shall use the distinctive subscript $f_{u\to v}$ to refer to the morphism variable $u \to v$ that is actually the morphism $f : |u| \to |v|$. This is necessary in order to define the domain and codomain of variables in a unique way. Since u and v are finite strings, the collection of morphism variables $u \to v$ remains a set.

If $F : \mathcal{M} \to \mathcal{M}'$ is a monoidal functor, then $\mathcal{A}F : \mathcal{AM} \to \mathcal{AM}'$ is the alphabet mapping ϕ by which $\phi A = FA$ and

$$\phi f_{u\to v} = Ff_{Fu\to Fv}$$

for every morphism $f : |u| \to |v|$ in $\mathcal{M}$, where F is extended to strings of objects in the obvious natural way.

With the above definition we have established the functor

$$\mathcal{A} : \mathbf{MonCat} \to \mathbf{Alph}$$

between the category of monoidal categories and that of ranked alphabets. Our aim is to provide a left adjoint $\mathcal{G}$ for the functor $\mathcal{A}$, when restricted to the subcategory **TraMon** of traced monoidal categories. In algebraic terms this amounts to constructing the traced monoidal category freely generated by some doubly ranked alphabet Σ. In order to keep the discussion simple, we shall assume that $O = \mathbb{N}$ that is, our categories are single-sorted. Then the free traced monoidal category generated by a doubly ranked alphabet is essentially the category of flowchart schemes as described in [1]. The only significant difference arises from the absence of the axiom:

$$Tr^U_{I,I} 1_U = 1_I,$$

where $1_U : U \to U$ denotes the identity morphism. In the single-sorted setting this axiom is equivalent to

$$\uparrow 1_1 = 1_0,$$

which was imposed as axiom S6 in [1]. (Read $\uparrow$ simply as $Tr^1_{1,1}$.) If this axiom is not present, then the policy with respect to the so called loop vertex in schemes changes as follows: every subexpression $\uparrow 1_1$ contributes a separate loop vertex $0 \to 0$ to the scheme being constructed. Thus, loop vertices multiply during the construction, as opposed to the policy applied in [8, 1] that there is a unique loop vertex in each scheme. (In other words, loop vertices do not multiply.) This straightforward change covers the whole impact of adding axiom $S6$ to the standard trace monoidal category axioms. The change is clearly visible e.g. in the category $(\mathbf{FdVect}_K, \otimes)$, where the sequence of morphisms

$$Tr^{2^0}_{2^0,2^0} 1_{2^0},\ Tr^{2^1}_{2^1,2^1} 1_{2^1},\ Tr^{2^2}_{2^2,2^2} 1_{2^2}, \ldots, Tr^{2^n}_{2^n,2^n} 1_{2^n}, \ldots$$

produces the sequence of elements

$$1,\ 2 = 1 + 1,\ 2^2 = 2 \cdot 2, \ldots, 2^n = 2 \cdot 2 \cdot \ldots \cdot 2, \ldots \text{ in } K.$$

The same dilemma, whether to multiply the loop vertices or not in the axiomatization of schemes, occurred already to Bloom and Ésik when writing the paper [8]. The author knows this from Zoltán himself, who told him in 1986 that they decided not to multiply the loop vertices because they only had algebraic theories in mind for possible interpretations. Otherwise the issue was trivial. Since Zoltán used to be the advisor of the author in previous years, the loop vertices did not multiply in the axiomatization [1] either.

Rather than giving the formal definition of Σ-schemes for a doubly ranked alphabet Σ, we just show an example scheme $S : 4 \to 1$ in Fig. 9. The scheme S has 4 input channels and 1 output channel. The alphabet Σ consists of the morphism variables $h : 2 \to 1$ and $g : 1 \to 1$. A variable occurrence in S is a box labelled by that variable. Boxes have numbered input and output ports (numbered from left to right), with as many input (output) ports as the domain (respectively, codomain) of the corresponding variable. The input channels have exactly one output port, while the output channels have a single input port. Each output port in the scheme is connected to exactly one input port, and every input port is the endpoint of a single edge coming from an output port. In addition, there are an arbitrary number of isolated loop vertices (none in our scheme S), which vertices have no input or output ports and are labelled by the special symbol $\perp$ not in Σ.

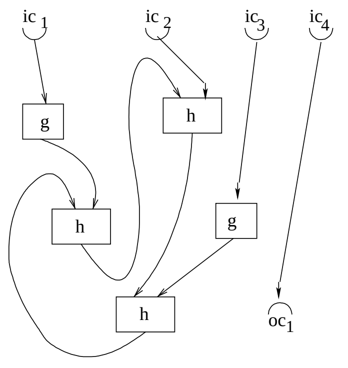

Figure 9: A Σ-scheme $4 \to 1$

Two Σ-schemes are *isomorphic* if they are isomorphic as directed graphs by an isomorphism ϕ which preserves the labeling of the boxes, and the input/output channels. Furthermore, ϕ must respect the numbering of the ports in boxes as well.

The traced monoidal category $Sch(\Sigma)$ of Σ-schemes uses the graph operations disjoint union for tensor, gluing schemes by their input/output channels for composition, and adding an edge from the first output channel to the first input channel for feedback, bypassing the pair of the connected channels themselves. Whenever feedback connects two channels that are already connected to each other in the opposite direction, a new instance of the loop vertex is created and added to the scheme. The interpretation of the identity and symmetry morphisms is straightforward, using straight lines connecting the input channels to the output ones. See [1, 6] for details. With this interpretation, the category $Sch(\Sigma)$ is that of isomorphism classes of Σ-schemes.

If Σ is not single-sorted, then the only modification to the above description is that ports are also labelled by the object variables in such a way that for every box labelled by morphism variable f, the sequence of object variables corresponding to the input (output) ports read from left to right is $dom(f)$ (respectively, $cod(f)$). Obviously, edges must respect the labeling of the ports. The scheme is $u \to v$ if the sequence of input (output) channel labels is u (respectively, v).

In order to interpret a scheme $S : x \to y$ in a traced monoidal category $\mathcal{C}$, one must first assign an object ϕA in $\mathcal{C}$ to each object variable A occurring in S, and concrete morphisms $\phi u \to \phi v$ in $\mathcal{C}$ to each morphism variable $u \to v$ occurring in S as a box label. Then represent S in normal form in the following way. Take the tensor (disjoint union) of the boxes in S, together with some straight lines, apply a suitable permutation from either side, and create the edges of S using feedback. Finally, copy this procedure at the level of semantics in the category $\mathcal{C}$. The result is a morphism $\phi x \to \phi y$, and it will not depend on the concrete syntactical normal form chosen, as long as we do not introduce a cycle of straight lines with feedback in the normal form. See again Fig. 7 in Section 3. This should be avoided by taking only a minimum number of straight lines in the normal form, unless there are loop vertices in the scheme. Each loop vertex $\perp_A$ counts as such a "looping" feedback, which corresponds semantically to $\uparrow^A 1_A$ with the chosen interpretation of the object variable A. Again, see [1, 6] for details.

We shall be interested in two more axioms in Section 5.

$$Acc \text{ (accessibility): } \; f = g \;\text{ for } f, g : I \to A, \text{ and}$$

$$Ter \text{ (termination): } \quad f = g \;\text{ for } f, g : A \to I.$$

By $Trace_{Biacc}$ we shall mean the set of traced monoidal category axioms extended by Acc and $Term$.

Notice that imposing the axioms Acc and $Term$ prompts the addition of constants $0_A : I \to A$ and $0^A : A \to I$ to the traced monoidal language. These constants must then satisfy the following further monoidal axioms:

$$0_A \circ f = 0_B \;\text{ for } f : A \to B,$$

$$0_A \otimes 0_B = 0_{A \otimes B},$$

$$(0_A \otimes 1_A) \circ c_{A,A} = 1_A \otimes 0_A,$$

and the dual counterparts of these axioms for 0^A. See [1, 13]. Regarding schemes as morphisms in the corresponding free category $Sch_{Biacc}(\Sigma)$, one must lift the condition that for each input port of a box (or output channel) there exists an edge arriving at that port, and dually, there need not exist an edge going out from any particular output port. (The ignored ports do not cease to exist, though, they just become idle.) All schemes must be biaccessible, however. Recall from [1, 8, 9] that a Σ-scheme S is *accessible* if each *box* (i.e. not necessarily the output channels) can be reached from at least one input channel through a directed path in S. Dually, S is *terminating* if from each box there exists a path to at least one output channel. Scheme S is *biaccessible* if it is both accessible and terminating. The axiomatization statement for $Sch_{Biacc}(\Sigma)$ as a free category follows from the yet more general axiomatization presented in [1].

5 The completeness result

In this section we take a closer look at the definition of a given monoidal category being complete for the traced monoidal category axioms extended by the axioms Acc and $Term$. We also show that already the initial single-sorted monoidal category satisfying the axioms $Trace_{Biacc}$, the category $(\mathbf{Pin}_{\mathbb{N}}, +)$ of finite partial injections over the objects $\mathbb{N}$ as a subcategory of $(\mathbf{Rel}_{\mathbb{N}}, +)$ is complete for $Trace_{Biacc}$.

The reader might have the impression that this statement is trivial, since $(\mathbf{Pin}_{\mathbb{N}}, +)$, being initial (with or without the additional constants $0_1 : 1 \to 0$ and $0^1 : 0 \to 1$), is present in the category of biaccessible schemes as a subcategory. While this is certainly true, it does not imply that $(\mathbf{Pin}_{\mathbb{N}}, +)$ is complete for $Trace_{Biacc}$. In general, there could be a lot of identities valid in the initial algebra of an equational class that are not valid in the free algebras of that class generated by several variables. For example, the initial D-algebra in [1], as a monoidal category, coincides with the initial algebraic theory, and it can be embedded in the single sorted monoidal category of cycle-free schemes with junctions allowed. (Add a constant $d : 2 \to 1$ to the single-sorted monoidal language, or $d_A : A \otimes A \to A$ in general, as the "diagonal". See [1, 13].) This category is the free D-algebra generated by its boxes, yet, it is very far from being an algebraic theory.

The motivation for our study is the observation made in [12] that the category $(\mathbf{FdVect}_K, \otimes)$ is complete for the traced monoidal axioms, whenever the field K has an infinite characteristic (i.e. the elements $1, 1+1, \ldots, 1+1+\ldots+1, \ldots$ are all distinct.) The authors of [12] rely on the following understanding of completeness.

> If two networks (schemes) have the same value under all interpretations over K, then they are isomorphic.

Our more formal understanding of interpretation and completeness is the following. Consider the doubly ranked alphabet $\Sigma = (O, M, r)$ (fixed for the rest of the paper) in which O is a countably infinite set and $M(p, q)$ is also countably infinite for each $p, q \in O^*$. In other words, we have as many object and morphism variables in Σ as we want in order to use them for labeling schemes. Then a *traced monoidal identity* is a pair $(\mathbf{p}, \mathbf{q})$ of Σ-schemes $u \to v$ for some $u, v \in O^*$. As usual, we write $\mathbf{p} = \mathbf{q}$ for the pair $(\mathbf{p}, \mathbf{q})$. Accordingly, by a $Trace_{Biacc}$ identity we mean a pair of biaccessible schemes.

Definition 2. A $Trace_{Biacc}$ identity $\mathbf{p} = \mathbf{q}$ is *valid* in a traced monoidal category $\mathcal{C}$ if for every alphabet mapping $\phi : \Sigma \to \mathcal{AC}$,

$$(\mathcal{G}\phi)\mathbf{p} = (\mathcal{G}\phi)\mathbf{q}.$$

Recall that $\mathcal{G} : \mathbf{Alph} \to \mathbf{TraMon}$ is the left adjoint of the forgetful functor $\mathcal{A}$. In the present case of biaccessible identities, take $\mathcal{G}$ to be the left adjoint of the appropriate counterpart of $\mathcal{A}$.

Definition 3. A traced monoidal category $\mathcal{C}$ satisfying the additional axioms Acc and $Term$ is *complete* for the traced monoidal axioms $Trace_{Biacc}$ if for every valid identity $\mathbf{p} = \mathbf{q}$ in $\mathcal{C}$, the biaccessible schemes $\mathbf{p}$ and $\mathbf{q}$ are isomorphic.

There is a slight problem with Definitions 2 and 3, which is highlighted by the following example.

Example Let Bi denote the axiom

$$Bi : \quad f = g \quad \text{for } f, g : I \to I,$$

and add it to the the traced monoidal ones to obtain the system $Trace_{Bi}$. Consider the obvious single-sorted traced monoidal category $(\mathbf{Bi}, +)$ of bijections $n \to n$ as a subcategory of $(\mathbf{Rel}_{fin}, +)$. Is it complete for $Trace_{Bi}$? One could immediately answer no, since e.g. Acc and Ter are valid in $(\mathbf{Bi}, +)$, yet they are not provable from $Trace_{Bi}$. This answer is not fair, however, because the axioms Acc and Ter

are just partially meaningful in $(\mathbf{Bi}, +)$, and when they are, they are equivalent to Bi.

The anomaly arises from the fact that categories are not simply multi-sorted algebras. Their hom-sets could be empty. Since we defined the alphabet Σ very generously, we must assign a morphism $|\phi p| \to |\phi q|$ to each morphism variable (box) $f : p \to q$ in Σ in order to create an alphabet mapping $\phi : \Sigma \to \mathcal{A}(\mathbf{Bi}, +)$. There will not be any, however, unless all object variables are mapped into $I = 0$ by ϕ. Indeed, if $A \neq I$ was mapped into say 1 (i.e. $\phi A = 1$), then – since the empty string $I = \epsilon$ goes automatically to 0 – the morphism variables $\epsilon \to A$ could not be mapped anywhere, for $\mathbf{Bi}(0, 1) = \emptyset$. One could (and must) get around this problem by including a "dummy" morphism in each hom-set of $(\mathbf{Bi}, +)$ and define the traced monoidal operations on these morphisms e.g. in the strict way (always resulting in the dummy morphism if either of the arguments is dummy). One may even extend $(\mathbf{Bi}, +)$ to the traced monoidal category of partial injections. The extension will not make the problem go away, though, because the axioms Acc and Ter would still remain valid in the extended categories but not in the quotient of $Sch(\Sigma)$ by Bi. This fact is already justifiable, however, since "bases are loaded" in the extended categories.

The above discussion shows that the anomaly of the Example is rooted in the polymorphic nature of the categorical language used to specify equational axioms (identities), and more analysis is required to provide a satisfactory explanation. For the moment, however, we shall rely on the straightforward solution of introducing the dummy morphisms if necessary, so at least we escape the contradiction arising from possibly empty hom-sets. Thus, the category $(\mathbf{Bi}, +)$ is presently not complete for $Trace_{Bi}$.

Theorem 1. *The category $(\mathbf{Pin}_{fin}, +)$ is complete for the axioms $Trace_{Biacc}$.*

Proof. Let $S, R : u \to v$ be two biaccessible Σ-schemes. We must find an interpretation ϕ of the alphabet Σ in $(\mathbf{Pin}_{fin}, +)$ that distinguishes these two schemes, provided that they are not isomorphic. Obviously, we can assume that both u and v are different from $I = \epsilon$. We can also assume, without loss of generality, that there exists an input-output path in at least one of S and R passing through at least one box. Indeed, otherwise it is trivial to separate S and R directly in $(\mathbf{Pin}_{fin}, +)$ by interpreting each object variable A as $\phi A = 1$ and each morphism variable $f : p \to q$ as $\phi f = 0^{\phi(p)} + 0_{\phi(q)}$, i.e., the totally undefined injection.

First let us assume that each box in S and R is labelled by the same morphism variable $f : p \to q$ in M. Since the schemes are biaccessible, $p \neq \epsilon$ and $q \neq \epsilon$. Let us spell out the trajectory of an input-output path in Σ-schemes. As in standard graph theory, it is an alternating sequence

$$q_0, e_1, (p_1, q_1), e_2, \ldots, (p_n, q_n), e_n, p_{n+1} \tag{3}$$

of vertices and edges, starting from and ending at a vertex (port), such that each edge e_i in the sequence is incident with the vertex (output port q_{i-1}) immediately preceding it and with the vertex (input port p_i) immediately following it. Remember that each port is numbered, and it is also labelled by an object variable in O.

For simplicity, the number assigned to the port of an input/output channel is the serial number of that channel.

For a path α of the form (3), $\psi(\alpha)$ will denote the "trace" of α:

$$(l_0, B_0), ((m_1, A_1), (l_1, B_1)), \ldots, ((m_n, A_n), (l_n, B_n)), (m_{n+1}, A_{n+1}).$$

In this sequence, $1 \leq l_0 \leq \mathit{length}(u)$ and $1 \leq m_{n+1} \leq \mathit{length}(v)$ identify the serial number of the input channel the output port of which is q_0 and that of the output channel identified by port p_{n+1}. The pair (m_i, A_i) $((l_i, B_i))$ consists of the serial number and object variable corresponding to the input port p_i (respectively, output port q_i). By definition, $B_i = A_{i+1}$ for every $0 \leq i \leq n$.

Now we turn to defining the separating interpretation ϕ. For all object variables $A \in O$, let $\phi(A) = k$, where k is a sufficiently large integer, the magnitude of which will be specified later. In this way we can think of a morphism variable $f : p \to q$ as a single-sorted one $f_k : k \cdot m \to k \cdot l$, where $m > 0$ and $l > 0$ are the length of p and q, respectively. The morphism $\phi f : k \cdot m \to k \cdot l$ will be the partial injection by which

$$(m_i - 1) \cdot k + i \mapsto (l_i - 1) \cdot k + i + 1 \tag{4}$$

for every $1 \leq i < k$ and appropriately chosen numbers $1 \leq m_i \leq m$, $1 \leq l_i \leq l$.

It is easy to visualize the partial injection $(\mathcal{G}\phi)S$ for scheme S using the Hungarian method discussed in Section 3. Every edge e in S counts as k parallel edges in the corresponding single-sorted scheme S_k, where each port is "multiplied" by k. Let M be the set of all edges in S_k. The set M becomes a matching if we consider S_k as a bipartite graph on the *ports* as vertices rather than one on the boxes. Add ϕf to S_k as edges *inside* the boxes, and let $S_k(\phi)$ denote the resulting bipartite graph. Intuitively, whenever the control reaches a box at input port m_i in "dimension" $i < k$, it will leave at output port l_i in dimension $i + 1$. Thus, $(\mathcal{G}\phi)S$ can be obtained by looking for "dual" augmenting paths in $S_k(\phi)$ with respect to matching M, that is, alternating paths starting and ending with M-positive edges incident with the set of leaves of $S_k(\phi)$ consisting of the input-output channels (ports). If such a path exists between input channel l in dimension i and output channel m in dimension j, then $(l, i) = (l - 1) \cdot k + i$ is mapped to $(m, j) = (m - 1) \cdot k + j$ according to $(\mathcal{G}\phi)S$. (Mind that an input channel is in fact an output port and an output channel counts as an input port.)

Observe that the procedure of calculating $(\mathcal{G}\phi)S$ by the Hungarian method would be exactly the same if we were to interpret S under relations, rather than partial injections.

What we need in order to finish the proof is an understanding of schemes S and R being isomorphic. It is a standard graph theory argument that S and R are isomorphic iff, for each input-output channel pair (l, m), whenever there exists a path α connecting l with m in the one scheme (say S), then there exists such a path β in R, too, so that $\psi(\alpha) = \psi(\beta)$. (Remember that ψ denotes the trace of a path.) Indeed, the numbered ports, together with the biaccessibility restriction, make the graph isomorphism test for S and R straightforward.

Now let us assume that S and R are not isomorphic. Then there exists an input-output path α in say S such that its trace $\psi(\alpha)$ is missing from the traces of paths in R. Clearly, the length of α remains under a fixed number that depends on the size of S and R only. Choose α to be shortest among such paths, and let N denote the length of α. For $k > N$, set the numbers m_i and l_i in (4) according to the parameters of α up to dimension N, and arbitrarily for $N \leq i \leq k$. It is clear that this choice of ϕf will distinguish S and R, since only $(\mathcal{G}\phi)S$ will take the input channel l_0 in dimension 1 to output channel m_N in dimension $N+1$.

If there are several morphism variables assigned as labels to the boxes of S and R, then the proof can be augmented by adding a distinctive "preamble" injection to the interpretation of each morphism variable in each dimension, and driving the control through this preamble. Details are straightforward and left to the reader. The proof is now complete. □

Corollary 1. *The traced monoidal category* $(\mathbf{Rel}_{fin}, +)$ *is complete for* $Trace_{Biacc}$.

Proof. Indeed, $(\mathbf{Pin}_{fin}, +)$ can be embedded in $(\mathbf{Rel}_{fin}, +)$ as a subcategory. Moreover, bases are loaded in both categories, that is, no hom-set is empty. Therefore any valid identity $\mathbf{p} = \mathbf{q}$ in $(\mathbf{Rel}_{fin}, +)$ is valid in $(\mathbf{Pin}_{fin}, +)$ as well. Consequently, by Theorem 1, $\mathbf{p}$ and $\mathbf{q}$ are isomorphic as biaccessible schemes. □

6 Conclusion

We have shown that the traced monoidal category $(\mathbf{Pin}_{fin}, +)$ of finite partial injections is complete for the extension of the traced monoidal axioms by two identities that reflect the biaccesible property of schemes as morphisms in the category freely generated by a given monoidal signature. The proof was a classical separation argument, showing that if two biaccessible schemes are not isomorphic, then there exists an interpretation in terms of finite partial injections which distinguishes them. Since $(\mathbf{Pin}_{fin}, +)$ is the initial single-sorted traced monoidal category satisfying the given axioms, our result holds for every traced monoidal category into which $(\mathbf{Pin}_{fin}, +)$ can be embedded as a sub-monoidal category, and in which the identities Acc and Ter are valid.

References

[1] Bartha, M. A finite axiomatization of flowchart schemes. *Acta Cybernetica*, 8(2):203–217, 1987.

[2] Bartha, M. An equational axiomatization of systolic systems. *Theoret. Comput. Sci.*, 55:265–289, 1987.

[3] Bartha, M. An algebraic model of synchronous systems. *Information and Computation*, 97:97–131, 1992.

[4] Bartha, M. Simulation equivalence of automata and circuits. In: Csuhaj-Varjú, E. and Ésik, Z. editors, *12th International Conference on Automata and Formal Languages*, Balatonfüred, Hungary, Local Proceedings, pages 86–99, 2008.

[5] Bartha, M. Equivalence relations of Mealy automata. In: Bordihn, H., Freund, R., Holzer, M., Kutrib, M., and Otto, F. editors, *First Workshop on Non-Classical Models of Automata and Applications*, Wroclaw, Poland, Proceedings: books@ocg.at 256, Austrian Computer Society, pages 31–45, 2009.

[6] Bartha, M. The monoidal structure of Turing machines. *Mathematical Structures in Computer Science*, 23(2):204–246, 2013.

[7] Bartha, M. (2011) Quantum Turing automata. In Löwe, B. and Winskel, G. editors, *8th International Workshop on Developments in Theoretical Computer Science (DCM 2012)*, pages 17–31, Electronic Proceedings in Theoretical Computer Science, 2014.

[8] Bloom, S.L. and Ésik, Z. Axiomatizing schemes and their behaviors. *J. Comput. System Sci.* 31:375–393, 1985.

[9] Bloom, S.L. and Ésik, Z. *Iteration Theories: The Equational Logic of Iterative Processes.* Springer-Verlag, Berlin, 1993.

[10] Căzănescu, V.E. and Ştefănescu, Gh. Towards a new algebraic foundation of flowchart scheme theory. *Fundamenta Informaticae*, 13:171–210, 1990.

[11] Ésik, Z. Identities in iterative and rational theories. *Computational Linguistics and Computer Languages*, 14:183–207, 1980.

[12] Hasegawa, M., Hofmann, M. and Plotkin, G. Finite dimensional vector spaces are complete for traced symmetric monoidal categories. In *Pillars of Computer Science: Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday*, Springer LNCS 4800, pages 367-385, February 2008.

[13] Hasegawa, M. Bialgebras in Rel. In *26th Conference on the Mathematical Foundations of Programming Semantics (MFPS XXVI), May 2010, Ottawa.* Electronic Notes in Theoretical Computer Science 265, pages 337-359, 2010.

[14] Joyal, A. and Street, R. The geometry of tensor calculus I. *Advances in Mathematics*, 88(1):55–112, 1991.

[15] Joyal, A., Street, R. and Verity, D. Traced monoidal categories. *Math. Proc. Camb. Phil. Soc.*, 119:447–468, 1996.

[16] Katis, P., Sabadini, N. and Walters, R.F.C. Feedback, trace, and fixed-point semantics. *Theoret. Informatics Appl.*, 36:181–194, 2002.

[17] Kelly, G. M. and Laplaza, M. L. Coherence for compact closed categories. *J. Pure Appl. Algebra*, 19:193–213, 1980.

[18] Lovász, L. and Plummer, M.D. *Matching Theory.* North Holland, 1986.

[19] Mac Lane, S. *Categories for the Working Mathematician.* Springer-Verlag, 1971.

[20] Mac Lane, S. and Paré, R. Coherence for bicategories and indexed categories. *J. Pure and Appl. Algebra*, 37:59–80, 1985.

[21] Malherbe, O., Scott, P.J., and Selinger, P. Partially traced categories. *Journal of Pure and Applied Algebra*, 216(12):2563–2585, 2011.

[22] Selinger, P. A note on Bainbridge's power set construction. Manuscript 10 pages, 1998.

[23] Selinger, P. Towards a quantum programming language. *Mathematical Structures in Computer Science* 14:527–586, 2004

[24] Selinger, P. A survey of graphical languages for monoidal categories. In Coecke (editor), *New Structures for Physics, Lecture Notes in Physics* 183, Springer-Verlag, Berlin, 2009.

[25] Ştefănescu, Gh. *Network Algebra.* Series in Discrete Mathematics and Theoretical Computer Science, Springer, Heidelberg, 2000.