From Natural Language to RDF Graphs with Pregroups

Antonin Delpeuch, Anne Preller

From Natural Language to RDF Graphs with Pregroups

Antonin Delpeuch
École Normale Supérieure
45 rue d'Ulm
75005 Paris, France
antonin.delpeuch@ens.fr

Anne Preller
LIRMM
161 rue Ada
34392 Montpellier Cedex 5, France
preller@lirmm.fr

Abstract

We define an algorithm translating natural language sentences to the formal syntax of RDF, an existential conjunctive logic widely used on the Semantic Web. Our translation is based on pregroup grammars, an efficient type-logical grammatical framework with a transparent syntax-semantics interface. We introduce a restricted notion of side effects in the semantic category of finitely generated free semimodules over $\{0, 1\}$ to that end. The translation gives an intensional counterpart to previous extensional models. We establish a one-to-one correspondence between extensional models and RDF models such that satisfaction is preserved. Our translation encompasses the expressivity of the target language and supports complex linguistic constructions like relative clauses and unbounded dependencies.

1 Introduction

There is a general agreement that Natural Language Processing has two aspects. One is syntax, rules how words are put together to form a grammatical string. The other is semantics, rules how meanings of strings are computed from the meanings of words. To this we add a third aspect, namely that semantics must include rules of logic how to compute the truth of the whole string from the truth of its parts.

The Resource Description Framework (RDF) (Hayes and McBride, 2004) is an artificial language that takes the intuitive form of knowledge graphs. Its semantics has the expressive power of the fragment of multisorted first order logic that uses conjunction and existential quantification only. This restricted expressive power limits the statements of natural language that can be interpreted as RDF graphs. Typically, statements with negation words are excluded.

We use pregroup grammars as the linguistic and mathematical tool to recognise and interpret natural language strings in RDF. Pregroup Calculus, also known as Compact Bilinear Logic (Lambek, 1993) (Lambek, 1999), is a simplification of Lambek's Syntactic Calculus, (Lambek, 1958). Pregroup grammars are based on a lexicon like all categorial grammars. Syntactical analysis consists in the construction of a proof (derivation) in the calculus.

All semantics proposed so far use functors from the lexical category of (Preller and Lambek, 2007) into some symmetric compact closed category. They include the compositional distributional vector space models of (Clark et al., 2008), (Kartsaklis et al., 2013) based on context and the functional logical models of (Preller, 2012).

We proceed as follows. Recalling that words and grammatical strings recognised by the grammar are represented by meaning-morphisms in the lexical category, we propose a "syntactic" functor from the latter into a symmetric monoidal category $\mathcal{C}_S$ of 'morphisms with side effects' over the category $\mathcal{C}$ of finite dimensional semimodules over the lattice $\mathbb{B} = \{0, 1\}$. The elements of the semimodule S identify with RDF graphs. The value of the syntactic functor at a statement is the RDF graph of the statement. The extensional models of logic are recast as "semantic" functors from the lexical category to $\mathcal{C}$. We associate to any semantic functor an RDF interpretation and show that a statement is true in the semantic functor if and only if the corresponding RDF graph is true in the RDF interpretation.

2 Preliminaries

2.1 RDF graphs

RDF is a widely-adopted framework introduced in (Hayes and McBride, 2004) as a standard for linked information representation on the Web. Informally, an RDF graph is a set of labeled links between objects, which represent statements concerning the linked objects.

Throughout this article we will simply consider that they are represented by strings of characters without spaces, written in a mono-spaced font: the entity *John* is denoted by the string `John`.

A link between two objects is a triple, made of one entity (the subject of the predicate), a property (the type of the link, also represented by a string) and a second entity (the object of the predicate). Graphically, we represent a triple as a directed property from the subject to the object, labeled by the predicate.

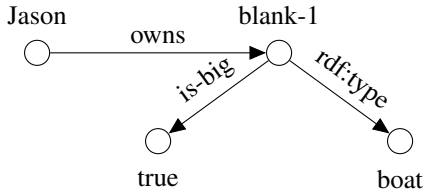
RDF allows to use nodes without labels, called blank nodes. Concretely, this means that we can always pick

a fresh node, named `blank-n` where `n` is some number, such that this node is not involved in any triple yet.

An example We can represent the sentence *John owns a big boat* using a fresh blank node for *a big boat*:

```
John owns blank-1
blank-1 rdf:type boat
blank-1 is_big true
```

Here, `rdf:type` is a special RDF predicate indicating the type of its subject. The graph representing this set of triples is



(a) A graph with a blank node

Recall that our goal is to translate natural language sentences to graphs. Graphs can indeed be seen as semantic representations of sentences, in the sense that they can be used to assign a truth value to a sentence, trough the notion of entailment (Hayes and McBride, 2004).

A graph $\mathcal{H}$ is an *instance* of $\mathcal{G}$ when $\mathcal{H}$ can be obtained from $\mathcal{G}$ by assigning labels to some blank nodes (possibly none of them). A graph $\mathcal{G}_0$ *entails* another graph $\mathcal{G}$ if an instance of $\mathcal{G}$ is a subgraph of $\mathcal{G}_0$. With these definitions, we can define true RDF graphs as the graphs entailed by some reference graph, storing all the true facts.

2.2 Pregroup grammars

A pregroup grammar (Lambek, 1999), consists of the free pregroup $\mathcal{C}(\mathcal{B})$ generated by a partially ordered set $\mathcal{B}$ and a lexicon $\mathcal{D}_\mathcal{B}$. By 'free pregroup' we mean the free compact closed category $\mathcal{C}(\mathcal{B})$ generated by $\mathcal{B}$ following the version given in (Preller and Lambek, 2007)[1]. The lexicon $\mathcal{D}_\mathcal{B}$ introduces the monoidal category $\mathcal{L}_\mathcal{B}$, freely generated by the inequalities and the lexical morphisms given by the lexicon. Lexical entries have "formal meanings" in the *lexical category*, the free compact closed category generated by $\mathcal{B}$ and the morphisms introduced by the words. They are analogue to the lambda-terms intervening in categorial grammars, (Steedman, 1996).

The working of pregroup grammars can be described without explicit use of category theory. The main result of this section however, the decomposition lemma, invokes properties of the graphs proved in (Preller and Lambek, 2007).

We start with the formal definition of a monoidal category followed by the condition it must satisfy to be compact closed.

Definition 1. *A category $\mathcal{C}$ is*

1. ***monoidal*** *if there is a bifunctor $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$, a distinguished object I, the* unit *of $\otimes$, satisfying $(A \otimes B) \otimes C = A \otimes (B \otimes C)$, $(f \otimes g) \otimes h = f \otimes (g \otimes h)$* [2]
2. ***compact closed*** *if it is monoidal and there are contravariant endofunctors $(\ )^r$ and $(\ )^\ell$, called* right adjoint *and* left adjoint*, such that for every object A and every morphism $f : A \to B$ there are morphisms $\eta_f : I \to A^r \otimes A$, the* name *of f, and $\epsilon_f : A \otimes B^r \to I$, the* coname *of f, satisfying for any $g : B \to C$*

$$(A \otimes B)^r = B^r \otimes A^r, \quad (A \otimes B)^\ell = B^\ell \otimes A^\ell$$
$$(\epsilon_f \otimes 1_C) \circ (1_A \otimes \eta_g) = g \circ f$$
$$(1_{A^r} \otimes \epsilon_g) \circ (\eta_f \otimes 1_{C^r}) = (f \circ g)^r \,.$$

3. ***symmetric*** *if it is monoidal and there is a natural isomorphism $\sigma_{AB} : A \otimes B \to B \otimes A$ such that $\sigma_{AB}^{-1} = \sigma_{BA}$.*

Recall that $\otimes$ is a bifunctor if and only if the following equalities are satisfied for all objects A, B and all morphisms $f_i : A_i \to B_i, g_i : B_i \to C_i, i = 1, 2$

$$\begin{array}{l} 1_A \otimes 1_B = 1_{A \otimes B} \\ \textit{Interchange Law} \\ (f_2 \circ f_1) \otimes (g_2 \circ g_1) = (f_2 \otimes g_2) \circ (f_1 \otimes g_1) \end{array} \tag{1}$$

The morphisms of $\mathcal{C}(\mathcal{B})$ and $\mathcal{L}_\mathcal{B}$ identify with graphs, which we now describe without invoking category theory.

The objects of $\mathcal{C}(\mathcal{B})$ are called *types*. They include the elements of $\mathcal{B}$, called *basic types*. For instance, the set $\mathcal{B}$ may include the basic types $\boldsymbol{s}$ for statements, $\boldsymbol{d}$ for determiners, $\boldsymbol{n}$ for noun phrases, $\boldsymbol{o}$ for direct objects with corresponding types in relatives clauses $\boldsymbol{r}$, $\hat{\boldsymbol{n}}$ and $\hat{\boldsymbol{o}}$. There is only one strict inequality $\boldsymbol{n} < \boldsymbol{o}$. Assimilating $\mathcal{B}$ to a category, we write $i_{\boldsymbol{ab}} : \boldsymbol{a} \to \boldsymbol{b}$ instead of $\boldsymbol{a} \leq \boldsymbol{b}$ and $1_{\boldsymbol{a}}$ for $i_{\boldsymbol{aa}}$. A *simple* type is an iterated left adjoint or right adjoint of a basic type $\ldots, \boldsymbol{a}^{\ell\ell} = \boldsymbol{a}^{(-2)}, \boldsymbol{a}^\ell = \boldsymbol{a}^{(-1)}, \boldsymbol{a} = \boldsymbol{a}^{(0)}, \boldsymbol{a}^r = \boldsymbol{a}^{(1)}, \boldsymbol{a}^{rr} = \boldsymbol{a}^{(2)}, \ldots$. The *iterator* of $t = \boldsymbol{a}^{(z)}$ is the integer z. An arbitrary type is a string of simple types separated by the symbol $\otimes$. In particular, the empty string is a type, denoted I.

The *lexicon* of a pregroup grammar consists of a set of pairs *word* : T where the type $T \in \mathcal{C}(\mathcal{B})$ captures the different grammatical roles a word may play. For

[1] Semantics requires more than one morphism between objects, whereas the partial preorder of the free pregroup of (Lambek, 1999) confounds all morphisms with identical domain and codomain.

[2] Strictly speaking, these equalities hold up to natural isomorphisms, but the coherence theorem of (Mac Lane, 1971) makes it possible to replace the isomorphisms by equalities without loss of generality.

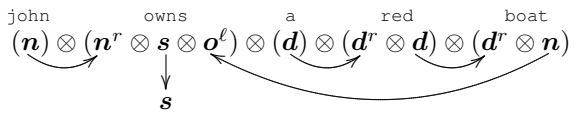
instance

$$
\begin{array}{ll}
\textit{proper name} & : \boldsymbol{n} \\
\textit{transitive verb} & : \boldsymbol{n}^r \otimes \boldsymbol{s} \otimes \boldsymbol{o}^\ell \\
\textit{transitive verb} & : \hat{\boldsymbol{n}}^r \otimes \boldsymbol{r} \otimes \boldsymbol{o}^\ell \\
\textit{transitive verb} & : \boldsymbol{n}^r \otimes \hat{\boldsymbol{o}}^r \otimes \boldsymbol{r} \\
\textit{determiner} & : \boldsymbol{d} \\
\textit{adjective} & : \boldsymbol{d}^r \otimes \boldsymbol{d} \\
\textit{countnoun} & : \boldsymbol{d}^r \otimes \boldsymbol{n} \\
\textit{relpronoun} & : \boldsymbol{n}^r \otimes \boldsymbol{n} \otimes \boldsymbol{r}^\ell \otimes \hat{\boldsymbol{n}} \\
\textit{relpronoun} & : \boldsymbol{n}^r \otimes \boldsymbol{n} \otimes \boldsymbol{r}^\ell \otimes \hat{\boldsymbol{o}}
\end{array}
$$

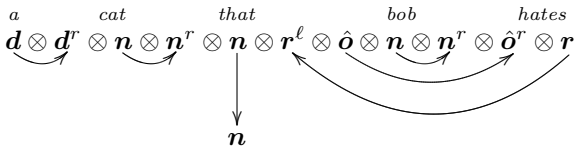
A pregroup grammars analyses a string of words by constructing a graph: choose an entry $w_i : T_i$ for each word w_i and align the types in the order of the words. Every simple type t_j of $T_1 \otimes \ldots \otimes T_n = t_1 \otimes \ldots \otimes t_m$ is (the label of) a node of the graph. Place non-crossing oriented underlinks such that the tail of an underlink is a basic type with an even number of iterations, say $t = \boldsymbol{a}^{(z)}$, where $z = \pm 2n$. If the head is to the left of the tail then it is the left adjoint $\boldsymbol{b}^{(z)\ell} = \boldsymbol{b}^{(z-1)}$, for some basic type $\boldsymbol{b} \geq \boldsymbol{a}$. If it is to the right then its a right adjoint $\boldsymbol{b}^{(z)r} = \boldsymbol{b}^{(z+1)}$. Complete the graph by repeating the nodes that are not head or tail of an underlink in a line below and adding a vertical link between corresponding nodes.

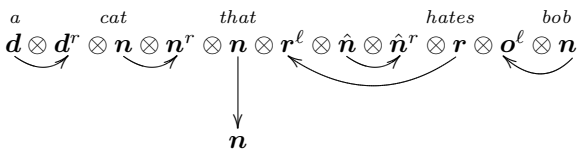
The string is said to be *grammatical* if exactly one simple type remains that is not the tail or head of an underlink and if it is a basic type. The resulting graph is called a *reduction* and denotes a unique morphism $r : T_1 \otimes \ldots \otimes T_n \to \boldsymbol{b}$ of $\mathcal{C}(\mathcal{B})$. More generally, graphs standing for morphisms align the domain above and the codomain below.

For instance, the graph below exhibits a reduction to the sentence type $\boldsymbol{s}$



The following two graphs are reductions to the noun phrase type $\boldsymbol{n}$. The first graph corresponds to the case where the relative pronoun is the object of the verb in the relative clause whereas it is the subject in the second graph

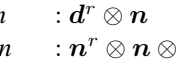




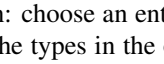
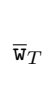
The meanings of words are also represented by graphs that correspond to morphisms of the lexical category $\mathcal{L}_\mathcal{B}$. In fact, every entry $w : T$ in the lexicon gives rise to a *meaning morphism* $\overline{\mathtt{w}}_T : I \to T$ and a *lexical morphism* $\mathtt{w}_T$ represented by the following oriented labelled graphs

$$\overline{\mathtt{w}}_{\boldsymbol{b}} \;=\; \overset{\mathtt{w}}{\underset{\boldsymbol{b}}{\downarrow}} \;=\; \mathtt{w}_{\boldsymbol{b}}$$

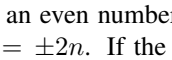
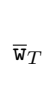
$T = \boldsymbol{a}^r \otimes \boldsymbol{b}$

$\overline{\mathtt{w}}_T =$  $\quad \mathtt{w}_T =$ 

$T = \boldsymbol{a}^r \otimes \boldsymbol{b} \otimes \boldsymbol{c}^\ell$

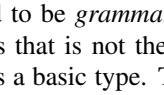
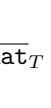
$\overline{\mathtt{w}}_T =$  $\quad \mathtt{w}_T =$ 

$T = \boldsymbol{a}^r \otimes \boldsymbol{c}^r \otimes \boldsymbol{b}$

$\overline{\mathtt{w}}_T =$  $\quad \mathtt{w}_T =$ 

If T has two factors that are basic types, there is besides the main lexical morphism w_T an auxiliary lexical morphism j_T.

$T = \boldsymbol{n}^r \otimes \boldsymbol{n} \otimes \boldsymbol{r}^\ell \otimes \boldsymbol{d},\ \boldsymbol{d} = \hat{\boldsymbol{n}}, \hat{\boldsymbol{o}}$

$\overline{\mathtt{that}}_T =$  

We omit the subscript T if this does not lead to confusion.

The nodes of the meaning graphs are the simple types of T. They form the lower line of the graph, the upper line is the empty string I. The corresponding morphism has domain I and codomain T. An overlink may have several tails but only one head. The tails of overlinks are right or left adjoints of basic types, the head is a basic type[3].

The *lexical category* $\mathcal{L}_\mathcal{B}$ generated by the lexicon $\mathcal{D}_\mathcal{B}$ is the compact closed category freely generated by $\mathcal{B}$, the symmetry morphisms $\sigma_{\boldsymbol{ab}}$ and the lexical morphisms introduced by $\mathcal{D}_\mathcal{B}$.

Strings of words also have meaning(s) in the lexical category. The *lexical meaning* of a grammatical string *word*$_1 \ldots$ *word*$_n$ *recognised by the reduction* $r : T_1 \ldots T_n \to \boldsymbol{b}$ is, by definition, the composite of the tensor product of the word meanings and the chosen reduction.

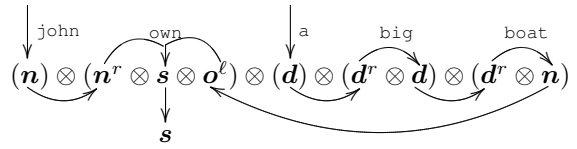
$$r \circ (\overline{\mathtt{word}_1}_{T_1} \otimes \ldots \otimes \overline{\mathtt{word}_n}_{T_n}) : I \to \boldsymbol{b}\,.$$

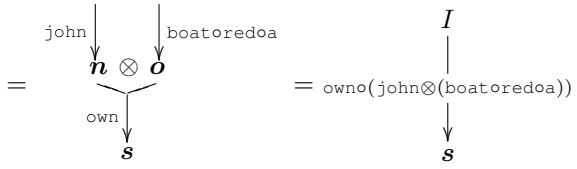
The meaning of a composite morphism $g \circ f$ can be simplified graphically. Stack the graph of f above the graph of g and walk the paths of the graph starting at a node that is not the head of a link until you arrive at a node that is not the tail of a link. Compose the

[3] "basic type" is replaced by simple type with an even iterator in the general case

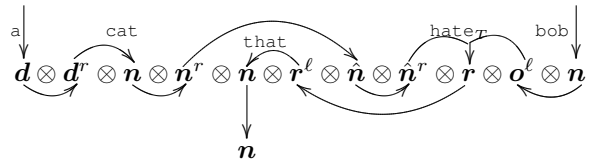
labels in the order they are encountered along the path. Replace the whole path by a single link labelling it by the composite label of the path. The resulting graph represents the morphism $g \circ f$, (Selinger, 2011).

For instance, the grammatical strings mentioned above above simplify to



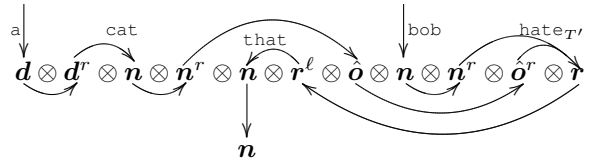


$T = \hat{\boldsymbol{n}}^r \otimes \boldsymbol{r} \otimes \boldsymbol{o}^\ell$



$\mathtt{that} \circ \mathtt{hate}_T \circ ((\mathtt{cat} \circ \mathtt{a}) \otimes \mathtt{bob})] : I \otimes I \to \boldsymbol{n}$

$T' = \boldsymbol{n}^r \otimes \hat{\boldsymbol{o}}^r \otimes \boldsymbol{r}$



$= \mathtt{that} \circ \mathtt{hate}_{T'} \circ ((\mathtt{cat} \circ \mathtt{a}) \otimes \mathtt{bob})] : I \otimes I \to \boldsymbol{n}$

All unlabelled links in the graph above correspond to inequalities of basic types. Inserting the strict inequalities at the appropriate place and applying the interchange law, we decompose the label into minimal building blocks that correspond one-to-one to the resources occurring in the RDF graph associated to the statement. For example,

$$\begin{aligned} &\mathtt{own} \circ (\mathtt{john} \otimes (\mathtt{boat} \circ \mathtt{big} \circ \mathtt{a})) \\ &= \mathtt{own} \circ (1_{\boldsymbol{n}} \otimes \mathit{in}) \circ (1_{\boldsymbol{n}} \otimes \mathtt{boat}) \circ (1_{\boldsymbol{n}} \otimes \mathtt{big}) \\ &\circ (\mathtt{john} \otimes \mathtt{a})\,. \end{aligned} \tag{2}$$

The expression after the equality symbol above is a composite of tensor products the factors of which are either inequalities between basic objects or lexical morphisms. Only the rightmost tensor product contains more than one lexical morphism. In fact, all factors are lexical morphisms with domain I.

The translation of statements to RDF graphs rests on the existence of this decomposition. Borrowing the terminology of RDF graphs, call any lexical morphism with domain I a *node word* and any other lexical morphism a *property word*.

Lemma 1 (Decomposition)**.** *Let $word_1 \dots word_n$ be a grammatical string with lexical morphisms $\mathtt{word}_1, \dots, \mathtt{word}_n$ and a reduction $r : T_1 \dots T_n \to \boldsymbol{b}$. Then there is an enumeration of the node words $\mathtt{word}_{i_1} : I \to \boldsymbol{b}_1, \dots, \mathtt{word}_{i_m} : I \to \boldsymbol{b}_m$ such that the lexical meaning of the string satisfies*

$$\begin{aligned} &r \circ (\overline{\mathtt{word}}_1 \otimes \dots \otimes \overline{\mathtt{word}}_n) \\ &= p_1 \circ \dots \circ p_{m'} \circ (\mathtt{word}_{i_1} \otimes \dots \otimes \mathtt{word}_{i_m})\,, \end{aligned}$$

where each p_k is either a tensor product of inequalities of basic types or a tensor product of inequalities and one property word $\mathtt{word}_{j_k}$. Moreover, $k < k'$ implies $j_k < j_{k'}$.

In particular, the meaning of the string belongs to the monoidal category generated by the lexicon

This is a straightforward consequence of the characterisation of morphisms as normal graphs in the free category, (Preller and Lambek, 2007) and the interchange law.

We map lexical morphisms to "unfinished" RDF triples

lexical morphism	RDF triple
$\mathtt{noun} : \boldsymbol{d} \to \boldsymbol{n}$	? rdf:type noun
$\mathtt{adjective} : \boldsymbol{d} \to \boldsymbol{d}$	? is-adjective true
$\mathtt{verb} : \boldsymbol{a} \otimes \boldsymbol{b} \to \boldsymbol{c}$	? verb ?
$\mathtt{determiner} : I \to \boldsymbol{d}$	blank ? ?, ? ? blank
$\mathtt{propername} : I \to \boldsymbol{n}$	propername ? ?, ? ? propername

(3)

The question marks designate unoccupied positions in the triple. Node words occupy either the subject or the object position, unary property words leave only the subject position open, binary property words occupy the centre position leaving subject and object position unoccupied.

3 The Translation

Let $\mathbb{B} = \langle \{0, 1\}, +, \cdot \rangle$ be the commutative semiring in which the addition, $+$, is the lattice operator $\vee$ and the multiplication, $\cdot$, the lattice operator $\wedge$ on $\{0, 1\}$.

The *semantic* category which hosts the RDF graphs and our models of grammatical strings is the category $\mathcal{C}$ of finitely generated semimodules over $\mathbb{B}$. It is compact closed satisfying $A^\ell = A = A^r$ for each object A. Every object A has a'canonical' basis $(e_i)_i$ such that every element $v \in A$ can be written uniquely as $v = \sum_i \alpha_i e_i$, where $\alpha_i \in \mathbb{B}$. We refer to elements of$A$ as *vectors*. Morphisms of $\mathcal{C}$ are maps that commute with addition and scalar multiplication.

We interpret RDF graphs as elements of a semimodule S determined by the RDF vocabulary, see below. The translation of grammatical strings is given by a functor from the monoidal category generated by the lexical morphisms into a *category of morphisms with side effects* mapping the decomposition of a grammatical string to a vector of S.

Let L be a set of *labels* that includes the property words and a 'very big', but finite number n_0 of labels

$\texttt{blank}_i$, for $i = 1, \ldots n_0$[4]. The other elements of L are node names and property names of an RDF vocabulary.

Define N as the semimodule over the semi-ring $\mathbb{B}$ freely generated by L and denote $e_{\texttt{label}}$ the basis vector of N corresponding to $\texttt{label} \in L$. We present RDF triples by basis vectors of $S = N \otimes N \otimes N$ and RDF graphs by sums of basis vectors of S, for instance $e_{\texttt{bob}} \otimes e_{\texttt{hate}} \otimes e_{\texttt{blank}} + e_{\texttt{blank}} \otimes e_{\texttt{rdf-type}} \otimes e_{\texttt{cat}}$. Hence, the vector sum models the union of RDF graphs.

We want to interpret the lexical morphisms such that they construct a triple when applied to an argument and add it to the vector of S already constructed. Composition of the category $\mathcal{C}$ alone is not powerful enough to achieve this. We define a new category $\mathcal{C}_S$ in which the entity *a big boat* will be denoted by the pair $(e_{\text{blank-1}}, e_{\text{blank-1}} \otimes e_{\text{rdf:type}} \otimes e_{\text{boat}} + e_{\text{blank-1}} \otimes e_{\text{is-big}} \otimes e_{\text{true}})$.

Therefore we switch from $\mathcal{C}$ to the monoidal category $\mathcal{C}_S$ below in which arrows have two components. The first component creates the triple and the second component adds the new triple to the graph as a 'side effect'.

Definition 2 (Category with Side Effects)**.** *Let $\{a_1, \ldots, a_m\}$ and $\{b_1, \ldots, b_n\}$ be the basis[5] of A and B. For any $p \in \mathcal{C}(A, S)$, $q \in \mathcal{C}(B, S)$, define $p \otimes_+ q \in \mathcal{C}(A \otimes B, S)$ as the unique linear map satisfying for an arbitrary basis vector $a_i \otimes b_j$ of $A \otimes B$*

$$(p \otimes_+ q) : a_i \otimes b_j \mapsto p(a_i) + q(b_j)\,.$$

The category $\mathcal{C}_S$ of morphisms with side effects in S has:

- *objects as in $\mathcal{C}$*
- *morphisms $(f, p) : A \to B$ where $f \in \mathcal{C}(A, B)$, $\mathrm{Ker}(f) = \{0\}$ and $p \in \mathcal{C}(A, S)$.*
- *arrows $1_A = (1_A, 0)$.*
- *an operation $\circ$ defined by $(f, p) \circ (h, q) = (f \circ h, p \circ h + q)$.*
- *an operation $\otimes$ on objects defined as in $\mathcal{C}$*
- *an operation $\otimes$ on arrows defined by $(f, p) \otimes (h, q) = (f \otimes h, p \otimes_+ q)$.*

Examples of morphisms in the first component are the symmetry $\sigma : N \otimes N \to N \otimes N$, $\pi_1, \pi_2 : N \otimes N \to N$ defined by $\sigma(a_i \otimes b_j) = b_j \otimes a_i$, $\pi_1(a_i \otimes b_j) = a_i$ and $\pi_2(a_i \otimes b_j) = b_j$.

The new operation $\otimes_+$ introduced above concerns the second component only. The morphism $p \otimes_+ q$ computes at $a_i \otimes b_j$ the union of the RDF graphs $p(a_i)$ and $q(b_j)$, computed separately by p and q.

As an illustration, consider the following morphisms of $\mathcal{C}_S$

$$m_{\texttt{john}} = (e_{\texttt{john}}, 0),\ m_{\texttt{blank}_i} = (e_{\texttt{blank}_i}, 0)$$

The arrow of a proper noun consists of the node representing this entity, e.g. e_{john}, paired with the empty graph represented by $0 \in S$. Choosing the empty graph means that nothing is said about this node. A similar remark holds for determiners.

$$m_{\texttt{big}} = (1_N, 1_N \otimes e_{\texttt{is-big}} \otimes e_{\texttt{true}}),$$

$$m_{\texttt{boat}} = (1_N, 1_N \otimes e_{\texttt{rdf-type}} \otimes e_{\texttt{boat}})$$

An adjective or a noun is a morphism that maps a node word to itself and to the empty slot in the corresponding triple. As a side effect, it adds this triple to the second component.

$$m_{\texttt{own}} = (1_N \otimes e_{\texttt{own}} \otimes 1_N, 1_N \otimes e_{\texttt{own}} \otimes 1_N)$$

A transitive verb is a morphism that maps an ordered pair of nodes to a triple making the first the subject and the second the object of the triple.

Compose the morphisms

$$\begin{aligned}
&m_{\texttt{big}} \circ m_{\texttt{blank}_i} = (e_{\texttt{blank}_i}, t_1)\\
&\quad t_1 = (e_{\texttt{blank}_i} \otimes e_{\texttt{is-big}} \otimes e_{\texttt{true}})\\
&m_{\texttt{boat}} \circ m_{\texttt{big}} \circ m_{\texttt{blank}_i} = (e_{\texttt{blank}_i}, t_2 + t_1)\\
&\quad t_2 = (e_{\texttt{blank}_i} \otimes e_{\texttt{rdf-type}} \otimes e_{\texttt{boat}})\\
&m_{\texttt{own}} \circ (m_{\texttt{john}} \otimes (m_{\texttt{boat}} \circ m_{\texttt{big}} \circ m_{\texttt{blank}_i})) =\\
&m_{\texttt{own}} \circ (e_{\texttt{john}} \otimes e_{\texttt{blank}_i}, 0 + t_2 + t_1) =\\
&(e_{\texttt{john}} \otimes e_{\texttt{own}} \otimes e_{\texttt{blank}_i}, t_3 + t_2 + t_1)\\
&\quad t_3 = e_{\texttt{john}} \otimes e_{\texttt{own}} \otimes e_{\texttt{blank}_i}
\end{aligned} \tag{4}$$

The effect of composition is to create a new triple on the left of the comma and to store it on the right.

Proposition 1. *The category with side effects $\mathcal{C}_S$ is a monoidal category.*

The proof of this proposition is given in appendix A.

Define a monoidal structure preserving functor $\mathcal{F}$ from the monoidal category generated by the lexical morphisms to $\mathcal{C}_S$ thus

- $\mathcal{F}(\boldsymbol{s}) = S = N \otimes N \otimes N$
- $\mathcal{F}(\boldsymbol{a}) = N$ if $\boldsymbol{a} \neq \boldsymbol{s}$
- $\mathcal{F}(1_{\boldsymbol{a}}) = \mathcal{F}(i_{\boldsymbol{a}\,\boldsymbol{b}}) = \mathcal{F}(j_{\boldsymbol{a}\,\boldsymbol{b}}) = 1_{\mathcal{F}}(\boldsymbol{a})$, for all basic types $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{B}$
- $\mathcal{F}(\texttt{that} : \boldsymbol{r} \to \boldsymbol{n}) = (1, 0)$
- $\mathcal{F}(\texttt{name} : I \to \boldsymbol{d}) = (e_{name}, 0)$
- $\mathcal{F}(\texttt{determiner} : I \to \boldsymbol{d}) = (e_{\texttt{blank}_i}, 0)$
- $\mathcal{F}(\texttt{word} : \boldsymbol{d}^r \otimes \boldsymbol{n}) = (1, 1 \otimes e_{\texttt{rdf-type}} \otimes e_{\texttt{word}})$
- $\mathcal{F}(\texttt{word} : \boldsymbol{d}^r \otimes \boldsymbol{d}) = (1, 1 \otimes e_{\texttt{is-word}} \otimes e_{\texttt{true}})$
- $\mathcal{F}(\texttt{word}_{\boldsymbol{n}^r \otimes \boldsymbol{s} \otimes \boldsymbol{o}^l}) = (1 \otimes e_{\texttt{word}} \otimes 1, 1 \otimes e_{\texttt{word}} \otimes 1)$
  $\mathcal{F}(\texttt{word}_{\hat{\boldsymbol{n}}^r \otimes \boldsymbol{r} \otimes \boldsymbol{o}^l}) = (\pi_1, 1 \otimes e_{\texttt{word}} \otimes 1)$
  $\mathcal{F}(\texttt{word}_{\boldsymbol{n}^r \otimes \hat{\boldsymbol{o}}^r \otimes \boldsymbol{r}}) = (\pi_2, (1 \otimes e_{\texttt{word}} \otimes 1) \circ \sigma)$.

Note that the morphisms in the example above satisfy $m_{\texttt{word}} = \mathcal{F}(\texttt{word})$. Computation (4) shows that $\mathcal{F}$ maps the lexical meaning of *john owns a big boat* to the three RDF triples t_1, t_2, t_3.

Lemma 2. *Let $word_1 \ldots word_n$ be a statement with corresponding lexical entries $word_1 : T_1 \ldots word_n : T_n$ and $r : T_1 \ldots T_n \to \boldsymbol{s}$ a reduction to the sentence type. Then second component of*

$$\begin{aligned}
&\mathcal{F}(r \circ (\overline{\texttt{word}_1} \otimes \ldots \otimes \overline{\texttt{word}_n})) =\\
&(f, t_1 + \cdots + t_m) \in \mathcal{C}_S(I, S)
\end{aligned}$$

[4] it suffices that n_0 exceeds the number of occurrences of determiners in the set of digitalised documents

[5] In $\mathcal{C}$, every object has a unique basis: the canonical one.

is a sum of RDF triples $t_i \in S$, for $k = 1, \ldots, m$. Moreover, t_k has the form (3) *determined by the property word $\mathtt{word}_{j_k}$ such that the unlabelled nodes are filled by node words of the statement.*

The proof is given in Appendix B.

Definition 3 (RDF Translation)**.** *The* RDF graph translating the statement $word_1 \ldots word_n$ *with reduction $r : T_1 \ldots T_n \to \boldsymbol{s}$ is the second component of $\mathcal{F}(r \circ (\overline{\mathtt{word}_1} \otimes \ldots \otimes \overline{\mathtt{word}_n}))$.*

For instance, the translation of the statement *John owns a big boat* is

$$e_{\mathtt{john}} \otimes e_{\mathtt{own}} \otimes e_{\mathtt{blank}_i} + e_{\mathtt{blank}_i} \otimes e_{\mathtt{is\text{-}big}} \otimes e_{\mathtt{true}} + e_{\mathtt{blank}_i} \otimes e_{\mathtt{rdf\text{-}type}} \otimes e_{\mathtt{boat}} ,$$

because

$$\mathcal{F}(\mathtt{own} \circ (\mathtt{john} \otimes (\mathtt{boat} \circ \mathtt{big} \circ \mathtt{a})) = m_{\mathtt{own}} \circ (m_{\mathtt{john}} \otimes (m_{\mathtt{boat}} \circ m_{\mathtt{big}} \circ m_{\mathtt{blank}_i})) .$$

The translation algorithm from text to RDF graphs starts with a parsing algorithm of pregroup grammars that chooses a type for each word of the statement and provides a reduction to the sentence type. Next it computes the decomposition of the formal meaning in the lexical category by 'yanking'. Finally it computes the RDF graph by applying the translation functor $\mathcal{F}$ to the decomposition. The parsing algorithm is cubic polynomial in the number of words. Decomposition is linear, because the number of links is proportional to the number of words. Finally, translation again is linear, because the sum of the number of property words and of the number of node words in the decomposition is proportional to the number of words.

4 $\mathcal{C}$-Models and RDF Interpretations

In this section, we establish the connection between the extensional models of meaning of (Preller, 2012) with the RDF interpretations of (Hayes, 2004) via the translation $\mathcal{F}$ from natural language to RDF graphs. We show that a statement is true in an extensional model if and only if the RDF graph computed by $\mathcal{F}$ is true in the RDF interpretation associated to the extensional model.

Choose an object U of $\mathcal{C}$, the 'universe of discourse' of the fragment of natural language. The basis vectors of U stand for individuals or concepts. Properties are represented by maps that test (n-tuples of) entities and let (part of) them pass if the test succeeds.

Let $1 \leq i_1 < \cdots < i_m \leq n$ be a strictly increasing sequence. A linear map $q : U_1 \otimes \ldots \otimes U_n \to U_{i_1} \otimes \ldots \otimes U_{i_m}$ is said to be a *selector* if for any basis vector $e_1 \otimes \ldots \otimes e_n \in U_1 \otimes \ldots \otimes U_n$

$$q(e_1 \otimes \ldots \otimes e_n) = e_{i_1} \otimes \ldots \otimes e_{i_m} \text{ or } q(e_1 \otimes \ldots \otimes e_n) = 0 .$$

We say that q *selects the i-th factor* if $m = 1$ and $i = i_1$ and that it is a *projector* if $m = n$. If the latter is the case then q is an idempotent and self-adjoint endomorphism, hence a 'property' in quantum logic, see (Abramsky and Coecke, 2004).

If the domain V and the codomain $U_{i_1} \otimes \ldots \otimes U_{i_m}$ are fixed then the selectors are in a one-to-one correspondence with the 'truth-value' functions $p : A \to \{\top, \bot\}$ on the set A of basis vectors of V related by the condition

$$p(a) = \top \text{ if and only if } q(a) \neq 0$$

for all $a \in A$.

Let $v = \sum_{l=1}^{k} a_{j_l}$ be an arbitrary vector of V. A selector $q : V \to W$ is said to be *true at v* if $q(a_{j_l}) \neq 0$, for $l = 1, \ldots, k$.

Lemma 3. *Selectors are closed under composition and the tensor product. Every identity is a selector.*

Let $p : V \to W$ and $q : W \to X$ be selectors and $v \in V$. Then $q \circ p$ is true at v if and only if p is true at v and q is true at $w = p(v)$.

Proof. The first assertion is straight forward. To show the second, assume that a is a basis vector for which $(q \circ p)(a) \neq 0$. Then $p(a) \neq 0$ because $q(0) = 0$. Hence $p(a)$ is a basis vector of W selected by p. The property now follows for an arbitrary vector from the definitions. □

Definition 4. *A compact closed structure preserving functor $\mathcal{M} : \mathcal{L}_{\mathcal{B}} \to \mathcal{C}$ is a* $\mathcal{C}$-model with universe U *if it satisfies*

- $\mathcal{M}(\boldsymbol{s}) = U \otimes U$
- $\mathcal{M}(\boldsymbol{a}) = U$ *for any basic type $\boldsymbol{a} \neq \boldsymbol{s}$*
- $\mathcal{M}(1_{\boldsymbol{a}}) = \mathcal{M}(i_{\boldsymbol{a}\,\boldsymbol{b}}) = \mathcal{M}(j_{\boldsymbol{a}\,\boldsymbol{b}}) = 1_{\mathcal{M}(\boldsymbol{a})}$*, for all basic types $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{B}$*
- $\mathcal{M}(\mathtt{that}) = 1_U$
- $\mathcal{M}(\mathtt{word}_{\boldsymbol{a}^r \otimes \boldsymbol{b}})$ *and* $\mathcal{M}(\mathtt{word}_{\boldsymbol{n}^r \otimes \boldsymbol{s} \otimes \boldsymbol{o}^l})$ *are projectors*
- $\mathcal{M}(\mathtt{word}_{\hat{\boldsymbol{n}} \otimes \boldsymbol{r} \otimes \boldsymbol{o}^l}) = \pi_1 \circ \mathcal{M}(\mathtt{word}_{\boldsymbol{n}^r \otimes \boldsymbol{s} \otimes \boldsymbol{o}^l})$
  $\mathcal{M}(\mathtt{word}_{\boldsymbol{n}^r \otimes \hat{\boldsymbol{o}}^r \otimes \boldsymbol{r}}) = \pi_2 \circ \mathcal{M}(\mathtt{word}_{\boldsymbol{n}^r \otimes \boldsymbol{s} \otimes \boldsymbol{o}^l}) \circ \sigma$
- $\mathcal{M}$ *maps determiners and proper names in the singular to basis vectors.*

The last condition guarantees that the interpretations of a transitive verb in a statement and in a relative clause are equal if the relative pronoun is the subject of the verb in the relative clause and that they differ only by the symmetry isomorphism if the relative pronoun is the object.

Definition 5. *A statement $word_1 \ldots word_n$ with reduction $r : T_1 \otimes \ldots \otimes T_n \to \boldsymbol{b}$ is* true in $\mathcal{M}$ *if $\mathcal{M}(r \circ (\overline{\mathtt{word}}_1 \otimes \ldots \otimes \overline{\mathtt{word}}_n) \neq 0$.*

Truth in a model can be reformulated in terms of selector truth with the help of Lemma 1.

Lemma 4. *Let $p_1 \circ \cdots \circ p_{m'} \circ (\mathtt{word}_{i_1} \otimes \ldots \otimes \mathtt{word}_{i_m})$ be the decomposition of the formal meaning of the statement $word_1 \ldots word_n$. Then $\mathcal{M}(p_l)$ is a selector and $\mathcal{M}(\mathtt{word}_{i_k})$ a vector in U, for $1 \leq l \leq m'$, $1 \leq k \leq m$.*

Moreover, the statement is true in $\mathcal{M}$ if and only if the selector $\mathcal{M}(p_l)$ is true at $\mathcal{M}(p_{l+1}) \circ \cdots \circ \mathcal{M}(p_{m'}) \circ (\mathcal{M}(\mathtt{word}_{i_1}) \otimes \ldots \otimes \mathcal{M}(\mathtt{word}_{i_m}))$ for $1 \leq l \leq m'$.

Proof. $\mathcal{M}$ maps the meaning of the string to $\mathcal{M}(p_1) \circ \cdots \circ \mathcal{M}(p_{m'}) \circ (\mathcal{M}(\texttt{word}_{i_1}) \otimes \ldots \otimes \mathcal{M}(\texttt{word}_{i_m}))$. Note that for $k = 1, \ldots, m'$ any factor of $\mathcal{M}(p_k)$ is the identity of U unless it is $\mathcal{M}(\texttt{word}_{j_k}) = q_k$. Thus $\mathcal{M}(p_k)$ is a tensor product of selectors. Hence both assertions follow from Lemma 3. □

Any $\mathcal{C}$-model $\mathcal{M}$ defines an RDF interpretation. The *vocabulary* consists of the basis vectors $e_{\texttt{label}} \in N$, see previous section. The set of property symbols is given by

$$P = \{e_{\textit{is-adjective}} : \textit{adjective} \in \mathcal{D}\} \cup \{\texttt{rdf:type}\} \cup \{e_{\textit{verb}} : \textit{verb} \in \mathcal{C}\}$$

Define an RDF interpretation $\mathrm{I}_{\mathcal{M}}$ as follows

- set of properties

$$\mathrm{IP} = \{\mathcal{M}(\texttt{word}) : e_{\texttt{word}} \in P\} \cup \{\texttt{rdf:type}\}$$

- set of resources

$$\mathrm{IR} = \mathrm{IP} \cup U \cup \{\texttt{true}\}$$

- mapping IS and its extension to blank nodes

$$\mathrm{IS}(e_{\texttt{word}}) = \mathcal{M}(\texttt{word}),$$
$$\mathrm{IS}(e_{\texttt{blank}_i}) = \mathcal{M}(\texttt{determiner}_i)$$

- mapping IEXT from IP into the power set of IR × IR

$$\mathrm{IEXT}(\texttt{rdf-type}) = \{\langle e, \mathcal{M}(\texttt{noun})\rangle : \mathcal{M}(\texttt{noun}) \text{ is true at } e\}$$
$$\mathrm{IEXT}(\mathrm{IS}(e_{\texttt{is-adjective}})) = \{\langle e, \texttt{true}\rangle : \mathcal{M}(\texttt{adjective}) \text{ is true at } e\}$$
$$\mathrm{IEXT}(\mathrm{IS}(e_{\texttt{verb}})) = \{\langle e_1, e_2\rangle : \mathcal{M}(\texttt{verb}) \text{ is true at } e_1 \otimes e_2\}.$$

Proposition 2. *A statement $word_1 \ldots word_n$ is true in a $\mathcal{C}$-model $\mathcal{M}$ if and only if every triple of its RDF translation G is true in the RDF interpretation $I_{\mathcal{M}}$*

The proof is given in Appendix B.

5 Conclusion

The modelling of natural language by RDF graphs remains limited by the expressivity of RDF. The latter goes way beyond the few examples presented here. So far, we have only considered the extensional branch of a word. Future work will take advantage of the distinction between a property and its extension in RDF to introduce the conceptual branch of a plural, which does not refer to an extension, e.g *Tom likes* books, or capture the difference between Eve and John *own a boat* and *Eve and John* are athletes.

Acknowledgments

This work was supported by the École Normale Supérieure and the LIRMM. The first author wishes to thank David Naccache, Alain Lecomte, Antoine Amarilli, Hugo Vanneuville and both authors the members of the TEXTE group at the LIRMM for their interest in the project.

References

Samson Abramsky and Bob Coecke. 2004. A categorical semantics of quantum protocols. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science*, pages 415–425.

Stephen Clark, Bob Coecke, and Mehrnoosh Sadrzadeh. 2008. A compositional distributional model of meaning. In W. Lawless P. Bruza and J. van Rijsbergen, editors, *Proceedings of Conference on Quantum Interactions*. University of Oxford, College Publications.

Patrick Hayes and Brian McBride. 2004. Rdf semantics. Technical report, Hewlett Packard Labs.

Patrick Hayes. 2004. Rdf semantics. Technical report, W3C Recommendation, W3C.

Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, Stephen Pulman, and Bob Coecke, 2013. *Reasoning about Meaning in Natural Language with Compact Closed Categories and Frobenius Algebras*. Cambridge University Press.

Joachim Lambek. 1958. The mathematics of sentence structure. *American Mathematical Monthly*, 65:154–170.

Joachim Lambek, 1993. *Substructural Logics*, chapter From categorial grammar to bilinear logic, pages 207–237. Oxford University Press.

Joachim Lambek. 1999. Type grammar revisited. In Alain Lecomte, editor, *Logical Aspects of Computational Linguistics*, volume 1582 of *LNAI*, pages 1–27, Heidelberg. Springer.

Saunders Mac Lane. 1971. *Categories for the Working Mathematician*. Springer.

Anne Preller and Joachim Lambek. 2007. Free compact 2-categories. *Mathematical Structures for Computer Sciences*, 17(1):1–32.

Anne Preller, 2012. *From Sentence to Concept: Predicate Logic and Quantum Logic in Compact Closed Categories*, pages 00–29. Oxford University Press.

Peter Selinger. 2011. A survey of graphical languages for monoidal categories. In *New Structures for Physics*, volume 813 of *Lecture Notes in Physics*, pages 289–233. Springer.

Mark Steedman. 1996. *Surface Structure and Interpretation*, volume 30 of *Linguistic Inquiry Monograph*. MIT Press, Cambridge, Massachusetts.

A Proof of proposition 1

Note first that composition and the tensor are well defined.

Assume $(f, p) : A \to B$ and $(g, q) : B \to C$. Then $q \circ f : A \to S$ and $p : A \to S$, so $q \circ f + p : A \to S$. Therefore $(g, q) \circ (f, p) = (g \circ f, q \circ f + p) : A \to C$ is well defined. Similarly, if $(f_i, p_i) : A_i \to B_i$ for

$i = 1, 2$ then $p_1 \otimes_+ p_2 : A_1 \otimes A_2 \to S$ and therefore $(f_1 \otimes f_2, p_1 \otimes_+ p_2) : A_1 \otimes A_2 \to B_1 \otimes B_2$ as required.

The operation $\otimes_+$ is associative on arrows. Indeed, let $(a_i)_i$, $(b_j)_j$ and $(c_k)_k$ be the basis of A, B and C, respectively. Then for $r : C \to S$

$$\begin{aligned}&((p \otimes_+ q) \otimes_+ r)(a_i \otimes b_j \otimes c_k)\\ =&(p \otimes_+ q)(a_i \otimes b_j) + r(c_k)\\ =&p(a_i) + q(b_j) + r(c_k)\\ =&(p \otimes_+ (q \otimes_+ r))(a_i \otimes b_j \otimes c_k)\,.\end{aligned}$$

Hence the tensor product on arrows of $\mathcal{C}_S$ is associative.

To show the interchange law (1), we need a lemma:

Lemma 5. *Let $p : C \to S, q : D \to S$ and $f : A \to C, g : B \to D$ with $\mathrm{Ker}(f) = \mathrm{Ker}(g) = \{0\}$. Then*

$$(p \otimes_+ q) \circ (f \otimes g) = (p \circ f) \otimes_+ (q \circ g)$$

Indeed, let $(a_i)_i$, $(b_i)_i$, $(c_i)_i$ and $(d_i)_i$ be the bases of A, B, C and D respectively. For all i and j, we decompose $f(a_i)$ on the basis $(c_i)_i$ and similarly for $g(b_j)$ on $(d_i)_i$:

$$f(a_i) = \sum_r c_{i_r} \text{ and } g(b_j) = \sum_s d_{i_s}$$

Each family of indices $(i_r)_r$ and $(i_s)_s$ is non empty, because $\mathrm{Ker}(f) = \mathrm{Ker}(g) = \{0\}$. Hence,

$$\begin{aligned}&((p \otimes_+ q) \circ (f \otimes g))(a_i \otimes b_j)\\ =&(p \otimes_+ q)(f(a_i) \otimes g(b_j))\\ =&(p \otimes_+ q)((\sum_r c_{i_r}) \otimes (\sum_s d_{i_s}))\\ =&(p \otimes_+ q)(\sum_{r,s} c_{i_r} \otimes d_{i_s})\\ =&\sum_{r,s} p(c_{i_r}) + q(d_{i_s})\\ =&\sum_r p(c_{i_r}) + \sum_s q(d_{i_s})\\ =&p(f(a_i)) + q(g(b_j))\\ =&((p \circ f) \otimes_+ (q \circ g))(a_i \otimes b_j)\end{aligned}$$

The fifth equality uses the fact that $1 + 1 = 1$ and the sum is non empty. This terminates the proof of the lemma.

Now let $(f_1, p_1) : A \to C$, $(f_2, p_2) : C \to E$, $(g_1, q_1) : B \to D$ and $(g_2, q_2) : D \to F$. We show first the following equality

$$(f_1 \otimes_+ g_1) + (f_2 \otimes_+ g_2) = (f_1 + f_2) \otimes_+ (g_1 + g_2). \quad (5)$$

Indeed, let $(e_i)_i$ and $(f_j)_j$ be the bases of A and B respectively. Then, for all i and j,

$$\begin{aligned}&((f_1 \otimes_+ g_1) + (f_2 \otimes_+ g_2))(e_i \otimes f_j)\\ =&(f_1 \otimes_+ g_1)(e_i \otimes f_j) + (f_2 \otimes_+ g_2)(e_i \otimes f_j)\\ =&f_1(e_i) + g_1(f_j) + f_2(e_i) + g_2(f_j)\\ =&((f_1 + f_2) \otimes_+ (g_1 + g_2))(e_i \otimes f_j)\,.\end{aligned}$$

Finally, the Interchange Law follows from (5) and the definitions thus

$$\begin{aligned}&((f_2, p_2) \otimes (g_2, q_2)) \circ ((f_1, p_1) \otimes (g_1, q_1))\\ =&(f_2 \otimes g_2, p_2 \otimes_+ q_2) \circ (f_1 \otimes g_1, p_1 \otimes_+ q_1)\\ =&((f_2 \otimes g_2) \circ (f_1 \otimes g_1), (p_2 \otimes_+ q_2) \circ (f_1 \otimes g_1) + (p_1 \otimes_+ q_1))\\ =&((f_2 \circ f_1) \otimes (g_2 \circ g_1), (p_2 \circ f_1) \otimes_+ (q_2 \circ g_1) + (p_1 \otimes_+ q_1))\\ =&((f_2 \circ f_1) \otimes (g_2 \circ g_1), (p_2 \circ f_1 + p_1) \otimes_+ (q_2 \circ g_1 + q_1))\\ =&(f_2 \circ f_1, p_2 \circ f_1 + p_1) \otimes (g_2 \circ g_1, q_2 \circ g_1 + q_1)\\ =&((f_2, p_2) \circ (f_1, p_1)) \otimes ((g_2, q_2) \circ (g_1, q_1))\,.\end{aligned}$$

Therefore $\mathcal{C}_S$ is a monoidal category.

B Proof of Lemma 2 and Proposition 2

Proof. Note that both $\mathcal{F}$ and $\mathcal{M}$ map inequalities of basic types to identities, hence also any atom without a lexical morphism to an identity. Let $\mathtt{word}_{j_l}$ be the property word occurring in p_l, say as the k_l-th factor, $q_l = \mathcal{M}(\mathtt{word}_{j_l})$ and $m_l = \mathcal{F}(\mathtt{word}_{j_l})$, for $l = 1, \ldots, m$. Then $\mathcal{M}(p_l) = 1_U \otimes \ldots q_l \ldots \otimes 1_U$ and $\mathcal{F}(p_l) = 1_N \otimes \ldots m_l \ldots \otimes 1_N$. Suppose that e_i is a basis vector of U and $e_{\mathtt{node}_i}$ a basis vector of N satisfying $e_i = I(e_{\mathtt{node}_i})$.

Use induction on $n-m-l$ and assume that $e_1 \otimes \ldots \otimes e_{r_l} = \mathcal{M}(p_{l+1}) \circ \cdots \circ \mathcal{M}(p_{n-m}) \circ (\mathcal{M}(\mathtt{node}_{i_1}) \otimes \ldots \otimes \mathcal{M}(\mathtt{node}_{i_m}))$ and $(e_{\mathtt{node}_1}, 0) \otimes \ldots \otimes (e_{\mathtt{node}_{r_l}}, 0) = \mathcal{F}(p_{l+1}) \circ \cdots \circ \mathcal{F}(p_{n-m}) \circ (\mathcal{F}(e_{\mathtt{node}_{i_1}}) \otimes \ldots \otimes \mathcal{F}(e_{\mathtt{node}_{i_m}}))$.[6]

Let t_l be the triple created when composing $\mathcal{F}(p_l)$ with $(e_{\mathtt{node}_1}, 0) \otimes \ldots \otimes (e_{\mathtt{node}_{r_l}}, 0)$. We want the show that t_l is true in $\mathrm{I}_\mathcal{M}$ if and only if $\mathcal{M}(p_l)$ is true at $e_1 \otimes \ldots \otimes e_{r_l}$.

Consider the case where $\mathtt{word}_{j_l} : \boldsymbol{d} \to \boldsymbol{d}$. The other cases are shown similarly. Recall that $m_l \circ (e_{\mathtt{node}_{k_l}}, 0) = (e_{\mathtt{node}_{k_l}}, t_l)$, with $t_l = e_{\mathtt{node}_{k_l}} \otimes e_{\mathtt{is\text{-}word}_{j_l}} \otimes e_{\mathtt{true}}$. Hence, $\mathcal{F}(p_l)((e_{\mathtt{node}_1}, 0) \otimes \ldots \otimes (e_{\mathtt{node}_r}, 0)) = ((e_{\mathtt{node}_1}, 0) \otimes \ldots (e_{\mathtt{node}_{k_l}}, t_l) \ldots \otimes (e_{\mathtt{node}_r}, 0) = (e_{\mathtt{node}_1} \otimes \ldots \otimes e_{\mathtt{node}_r}, t_l)$. Then t_l is true in $\mathrm{I}_\mathcal{M}$ if and only if $(\mathrm{I}(e_{\mathtt{node}_{k_l}}), \mathtt{true}) \in \mathrm{IEXT}(\mathrm{I}(e_{\mathtt{is\text{-}word}}))$ if and only if q_l is true at e_{k_l}, by definition of I. On the other hand, $1_U \otimes \ldots q_l \ldots \otimes 1_U$ is true at $e_1 \otimes \ldots e_{k_l} \ldots \otimes e_r$ if and only if q_l is true at e_{k_l}. If that is the case then $\mathcal{M}(p_l)(e_1 \otimes \ldots e_{k_l} \ldots \otimes e_r) = e_1 \otimes \ldots e_{k_l} \ldots \otimes e_r$. □

[6] 0 can be replaced by any vector of S without changing the proof.