Formal Models of Analogical Proportions

Nicolas Stroppa
National Centre for Language Technology
Dublin City University
Glasnevin, Dublin 9
nstroppa@computing.dcu.ie

François Yvon
GET/ENST & LTCI, CNRS UMR 5141
46 rue Barrault - F-75013 Paris
yvon@enst.fr

Résumé

Modèles Formels de Proportions Analogiques

Les tâches classiques du traitement automatique des langues naturelles (TALN) s'appuient, dans un nombre grandissant de contextes opérationnels, sur des mécanismes d'apprentissage capables d'extraire automatiquement les régularités linguistiques à partir de corpus annotés. Parmi ces mécanismes, l'apprentissage par analogie se distingue par l'exploitation systématique, dans un cadre d'apprentissage symbolique, de relations de proportionnalité formelle entre les exemples de la base d'apprentissage.

Dans ce rapport, nous proposons une définition générale pour ces relations de proportionnalité, qui s'appuie sur un cadre algébrique générique et se décline pour un grand nombre de représentations usuellement employées en TALN : séquences, structures de traits, arbres étiquetés, etc. Pour chacune de ces structures, nous introduisons également des algorithmes permettant de résoudre les deux problèmes principaux qui sont la validation de relation de proportionnalité et le calcul du quatrième terme inconnu d'une proportion.

Abstract

Formal Models of Analogical Proportions

Natural Language Processing (NLP) applications rely, in an increasing number of operational contexts, on machine learning mechanisms which are able to extract, in an entirely automated manner, linguistic regularities from annotated corpora. Among these, analogical learning is characterized by the systematic exploitation, in a symbolic machine learning apparatus, of formal proportionality relationships that exist between training instances.

In this paper, we propose a general definition of these proportionality relationships, based on a generic algebraic framework. This definition is specialized to

handle a number of representations that are commonly encountered in NLP applications, such as words over a finite alphabet, feature structures, labeled trees, etc. In each of these cases, we provide and discuss algorithms for answering the two main computational challenges posed by proportionality relationships: the validation of a proportion and the computation of the fourth term of a proportion.

1 Introduction

An *analogical proportion*[1] is a relationship between four entities (x, y, z, t), denoted by $x : y :: z : t$ and which reads "*x* is to *y* what *z* is to *t*". An example of such a proportion is:

"*planets* are to the *sun* what *electrons* are to the *atomic nucleus*".

The ability to dynamically perceive and establish such analogical relationships is often recognized to lie at the core of human cognition [Indurkhya, 1992, Gentner et al., 2001, French, 2002]. Tests involving verbal or geometric analogies constitute one basic ingredient of many IQ tests such as, for instance, the SAT college entrance test [Hoffman, 1995].

These kinds of proportions have been extensively studied in Artificial Intelligence, mostly in the context of Analogical Reasoning and Problem Solving (see e.g. [Evans, 1968, Hofstadter and the Fluid Analogies Research Group, 1995, Wilson et al., 1995, Jani and Levine, 2000, Schmid et al., 2003] to name just a few). The main focus of these studies is the dynamic process of analogy-making which relates situations which, while being apparently very different, share a set of common high-level relationships. In the above mentioned example, both the solar system and the atom involve the rotation of one object around a centre, suggesting that the causes of these movements could be similar: the identification of the analogical proportion thus makes possible the transfer of knowledge from the solar system domain to the atom domain.

The notion of analogical proportion also has a long history in linguistics. It was revitalized in the early 20$^{\text{th}}$century by the work of de Saussure [1916], for whom proportions are the basis of analogical creations, a process involved in the construction of novel words. Proportions are also prevalent in modern theories of morphology, such as word-based morphology and related approaches [Matthews, 1974, Anderson, 1992, Stump, 2001, Blevins, 2006].[2]In these models, the analysis of surface word forms is based primarily on their systematic *relationships* with other full forms. This is in sharp contrast with morphemic approaches of morphology, which decompose word forms into minimal units such as morphemes. An example proportion between word forms is:

"*believe* is to *unbelievable* what *forgive* is to *unforgivable*".

The (formal) proportion between these surface forms denotes here a deeper relationship between linguistic entities: the proportion also holds for the morphological features (part-of-speech, genre, number, etc.) that are associated with them. Moreover, given the morphological features of the first three forms, it is possible to infer the morphological features of the fourth one by solving the *analogical equation*[3]

"*M(believe)* is to *M(unbelievable)* what *M(forgive)* is to *?*",

where $M(u)$ denotes the bundle of morphological properties associated with u.

Based on this observation, memory-based learning devices have been proposed to solve various Natural Language Processing (NLP) tasks such as automatic word pronunciation [Yvon, 1999], morphological analysis [Lepage, 1999a, Pirrelli and Yvon,

[1]These proportions correspond to the Aristotelian [Aristote] notion of Analogy.

[2]This notion has also been invoked in the field of syntax [Bloomfield, 1970, Itkonen and Haukioja, 1997, Lavie, 2003], even though the use of analogy in this domain has been limited by the generative claims about the "inadequacy" and "unclarity" of this notion [Chomsky, 1975].

[3]This kind of equation is also termed *computation of the fourth (proportional) term* [de Saussure, 1916] or *analogical deduction* [Blevins, 2006].

1999] and syntactic analysis [Lepage, 1999b]. These devices require the identification of *formal* analogical proportions between linguistic representations.[4] From a computational perspective, two key procedures need to be implemented: a *validation* procedure, which assesses the validity of a proportion; and a *resolution procedure*, which computes the fourth term of an analogical equation. These attempts have been limited by the types of representations they were able to handle: in most cases we know of, proportions were only defined on finite sequences over a finite alphabet, using a restricted or ad-hoc definition of the notion of proportion [Yvon, 1997, Lepage, 1998].

The main contribution of this paper is thus to establish a general and consistent framework for the notion of analogical proportion, which is necessary to widen the scope of application of these learning devices and make them applicable for a wider range of linguistic representations and NLP tasks. We therefore present a general model for analogical proportion, which is applicable to several algebraic structures, including free monoids and sets of trees. We show that this framework is a "natural" generalization of extant definitions of proportions between simple objects, and explain how this framework can be instantiated to yield a suitable computational model for complex and/or structured representations such as words, trees, feature structures, and languages, which are all commonly used in NLP applications. This contribution has both a methodological and a computational aspect: in addition to a sound definition of proportions, we are also interested in the algorithmic complexity of the associated validation and resolution procedures.

The potential of analogy-based learners using these procedures has already been studied experimentally, for example in [Stroppa, 2005, Stroppa and Yvon, 2005, 2006]. In this paper, we deliberately focus on the theoretical foundations of the learning process, i.e. proportions. Note that both Stroppa [2005] and Stroppa [2005] report experiments carried out on phonology and morphology learning tasks, in which one has to predict a pronunciation or a morphological analysis given an input word form. In the case of pronunciation, the output to predict is a sequence (of phonemes); in the case of morphological analysis, one may predict a (morphological) tree or a feature structure (of morphological features). These experiments demonstrate the ability of the approach to deal directly with various structured data while achieving state-of-the-art results; they are based on ALANIS[5] (A Learning-by-Analogy Inferencer for Structured data), the tool we developed to implement the theoretical framework that we describe here.

The remainder of this paper is organized as follows. To begin with, we introduce in Section 2 several examples of proportions that we want to model. In Section 3, we establish a general definition of analogical proportions, suitable for the following algebraic structures: semi-groups, free monoids, and lattices. This definition readily yields a suitable model for words, sets, and feature structures. The case of proportions between languages and labeled trees are addressed in Sections 4 and 5 respectively. We conclude and discuss avenues for further research in Section 6.

[4] These works, and others, have also been used to support the view that some of the cognitive processes involved in the recognition of words, or in the letter-to-sound mapping, could be modeled through some kinds of analogical process. We will not address here the cognitive plausibility of analogy-based language processing, but rather concentrate on the algorithmic aspects of the computation of proportion.

[5] This tool relies on the automata library VAUCANSON [Lombardy et al., 2004], which makes use of generic programming and static polymorphism [Régis-Gianas and Poss, 2003], allowing for both adaptability and efficiency.

2 Examples of analogical proportions

In this Section, we present some examples of proportions between linguistic representations that we want to model. We start with the case of words, then consider the case of trees, sets, and feature structures, using examples borrowed or adapted from [Matthews, 1981, Lepage, 2003, Yvon, 2003].

2.1 Analogical proportions between words

Word forms occurring in texts or lexicons are represented using strings, i.e. sequences over some finite alphabets. We want to model the following examples of proportions between word forms in English:

wolf : wolves :: leaf : leaves,
reader : unreadable :: doer : undoable,
unhappy : unhappiest :: dirty : dirtiest,

in Latin:

oratorem : orator :: honorem : honor,
([speaker, accusative] : [speaker] :: [honor, accusative] : [honor]),

and in Arabic:

ArsAlA : mursilin :: AslAmA : muslimin
([They (two) sent] : [senders] :: [They (two) became Muslims] : [Muslims]).

Proportions between word forms are often a sign of deeper relationships between the entities represented by these forms. Proportions can also be misleading, as in for instance the coincidental proportion $ear : early :: dear : dearly$.[6] Conversely, many morphological relationships between lexical entries do not translate into a proportion between their orthographic or phonetic representations: *slept* is the past tense form of *to sleep*, just as *gave* is the past tense form of *to give*; yet we do not observe a proportion between these four word strings.

As is obvious from these examples, our main focus is on *syntactic* (formal) proportions, i.e. proportions in which a word form is viewed as a meaningless sequence of letters (or more generally, of symbols over a finite set). We therefore do not recognize semantic relationships such as "*cow* is to *calf* what *mare* is to *foal*" as a valid proportion *over words*. This does not mean that our model is unable to handle these proportions, as will be demonstrated shortly, but rather that these relationships hold at a different level of representation than mere word strings.

2.2 Analogical proportions between trees

Trees are commonly used in linguistic representations, to make explicit the internal structure of complex words or sentences. Trees also constitute a natural representation of first order predicates, used in the representation of the semantics of linguistic entities. An example of proportion between syntactic trees, adapted from Matthews [1974] and which shows active/passive opposition, is displayed in Figure 1.

[6] Let us assume that we want to find the morphological analysis $M(dearly)$ of the unknown word form $dearly$. In this case, the misleading proportion $ear : early :: dear : dearly$ is actually not a problem for the analogical learners mentioned in the Introduction. Indeed, the resolution procedure, whose goal is to find a solution to the analogical equation $M(ear) : M(early) :: M(dear) :?$, will fail (this equation between morphological features has no solution), and no analysis will be proposed based on this proportion.

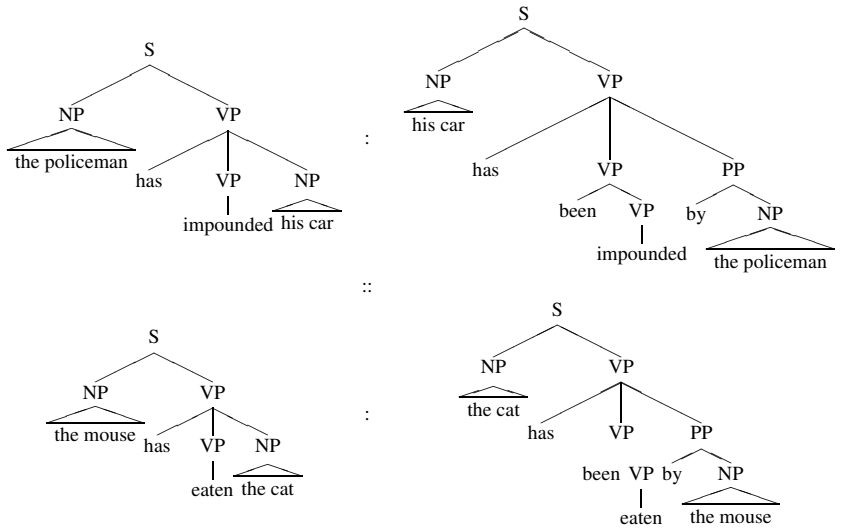


Figure 1: Analogical proportion between syntactic trees (adapted from Matthews [1974])

Such examples suggest that proportions may have the potential to handle some complex syntactic relationships between utterances, subject to the choice of an appropriate representation of these utterances (here, a representation that makes explicit the structural dependencies between words and phrases). Indeed, if we only consider the surface representations of these sentences (viewed as mere sequences of characters or words), the proportion ceases to hold. The ability to handle structural proportions such as the one in Figure 1 is thus an essential step to make analogical learning devices such as the one introduced in [Pirrelli and Yvon, 1999] applicable for Parsing or Example-Based Machine Translation tasks.

Tree representations can also be used to represent first-order logic predicates. We can also model the analogy "*cow* is to *calf* what *mare* is to *foal*", assuming a first-order representation of the meaning of these words (see Figure 2).

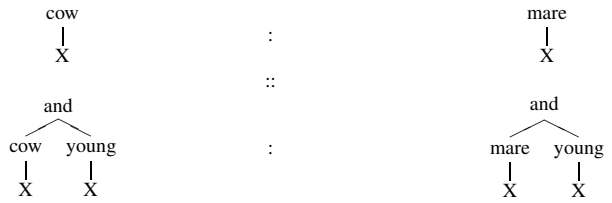


Figure 2: Analogical proportion between logical representations

2.3 Analogical proportion between sets

Classes are often used to represent discrete properties of a finite collection of linguistic entities. For example, a classical approach to the description of phonological systems identifies subclasses (subsets) of phonemes. Each class contains phonemes that share one or several properties: the class of consonants, of vowels, of stops, of consonant stops, etc. In this context, analogical proportions between sets of phonemes can be used to model systematic oppositions between these properties. For instance, the proportion displayed in Figure 3 relates stops (x), sibilants (t), voiced (y) and unvoiced (z) occlusive consonants.

$$x = \{ /b/, /p/, /t/, /d/, /k/, /g/ \} \quad : \quad y = \{ /b/, /d/, /g/, /z/, /v/, /ʒ/ \}$$
$$::$$
$$z = \{ /p/, /t/, /k/, /s/, /f/, /ʃ/ \} \quad : \quad t = \{ /s/, /z/, /f/, /v/, /ʃ/, /ʒ/ \}$$

Figure 3: Analogical proportion between sets of phonemes

2.4 Analogical proportion between feature structures

Systems of binary oppositions such as the one used in phonology can also be represented using feature structures, where each phoneme is associated with a vector of binary features. There is one feature for each subset of phonemes; a feature takes the value $+$ whenever the phoneme belongs to the related set. Using this representation, the fact that the opposition between /t/ and /d/ on the one hand, and /s/ and /z/ on the other hand is analyzed in terms of voiced/unvoiced alternation can be modeled by the proportion between the corresponding feature sets (see Figure 4). Binary feature structures are also used in semantic representations: Figure 5 gives another view on the proportion "*cow* is to *calf* what *mare* is to *foal*".

$$/t/ = \begin{bmatrix} consonantal : + \\ vocalic : - \\ voiced : - \\ anterior : + \\ coronal : + \\ continuant : - \\ strident : - \end{bmatrix} \quad : \quad /d/ = \begin{bmatrix} consonantal : + \\ vocalic : - \\ voiced : + \\ anterior : + \\ coronal : + \\ continuant : - \\ strident : - \end{bmatrix}$$
$$::$$
$$/s/ = \begin{bmatrix} consonantal : + \\ vocalic : - \\ voiced : - \\ anterior : + \\ coronal : + \\ continuant : + \\ strident : + \end{bmatrix} \quad : \quad /z/ = \begin{bmatrix} consonantal : + \\ vocalic : - \\ voiced : + \\ anterior : + \\ coronal : + \\ continuant : + \\ strident : + \end{bmatrix}$$

Figure 4: Analogical proportion between feature structures

Finally, feature structures are routinely used in NLP to represent syntactic structures. In the example of Figure 6, a proportion involving four such representations is displayed. Note that, in contrast to the two examples above (cf. Figures 4 and 5), the structures involved in this proportion are *complex*: the value of a feature can be another

$$\begin{array}{ccc} [cow/bull, female, mature] & : & [cow/bull, young] \\ & :: & \\ [horse, female, mature] & : & [horse, young] \end{array}$$

Figure 5: Mare and cows, again!

feature structure. The indexes occurring in these structure denote reentrancy, i.e. the fact that two features have a substructure in common.

$$\begin{array}{ccc} \left[\begin{array}{l} SUBJ : \left[\begin{array}{l} [1]\mathbf{agr} \\ PERS : [\,\mathbf{1st}\,] \\ NUM : [\,\mathbf{sing}\,] \end{array}\right] \\ PRED : [\,[1]\,] \end{array}\right] & : & \left[\begin{array}{l} SUBJ : \left[\begin{array}{l} [1]\mathbf{agr} \\ PERS : [\,\mathbf{3rd}\,] \\ NUM : [\,\mathbf{sing}\,] \end{array}\right] \\ PRED : [\,[1]\,] \end{array}\right] \\ :: & & \\ \left[\begin{array}{l} SUBJ : \left[\begin{array}{l} [1]\mathbf{agr} \\ PERS : [\,\mathbf{1st}\,] \\ NUM : [\,\mathbf{plur}\,] \end{array}\right] \\ PRED : [\,[1]\,] \end{array}\right] & : & \left[\begin{array}{l} SUBJ : \left[\begin{array}{l} [1]\mathbf{agr} \\ PERS : [\,\mathbf{3rd}\,] \\ NUM : [\,\mathbf{plur}\,] \end{array}\right] \\ PRED : [\,[1]\,] \end{array}\right] \end{array}$$

Figure 6: Analogical proportion between complex feature structures

2.5 Summary

We have introduced above a number of examples of proportions between linguistic representations. Being able to identify these proportions is a requirement for the application of some analogical learners to NLP tasks [Pirrelli and Yvon, 1999, Stroppa, 2005]. From the examples we gave, it should be clear that we are interested in modeling *formal* proportions, i.e. proportions that can be identified solely using the forms of the representations and the nature of the algebraic structure underlying them. Moreover, we aim at providing a consistent framework that would be applicable to the various types of representations we introduced. Note that we are also interested in other types of structures, such as finite languages and directed acyclic graphs (which can represent syntactic dependency graphs).

3 Analogical proportions on algebraic structures

In this section, we introduce a formal definition of the notion of analogical proportion. We start with a general definition, which is then specialized and simplified for several algebraic structures, notably semi-groups, free monoids, groups and lattices. For each of these structures, we also discuss the computational aspects of the basic algorithms which respectively validate a proportion and compute the fourth term in a proportion.

3.1 A generic framework

How can we define the notion of analogical proportion in such a way that all the previous examples are covered? To start with, let us reconsider one of the examples given in

Section 2, which we want to model. This example involves the four graphical forms: (*unhappy*, *unhappiest*, *dirty*, *dirtiest*). We can observe that the 1st and the 2nd forms share a prefix (*unhapp-*), so do the 3rd and the 4th forms (*dirt-*). Moreover, the 1st and the 3nd forms share a suffix (*-y*), so do the 2nd and the 4th forms (*-iest*). Each form can thus be analyzed with a prefix+suffix decomposition.

This analysis suggests that the notion of analogical proportion between four terms is related to the ability to *decompose* each term into several smaller fragments (two fragments in the example above) that can be exchanged, i.e. that *alternate*. This intuition can be formalized as follows. Let U be a semigroup, i.e. a set equipped with an internal associative law $\oplus$. In order to express the notion of decomposition, we first introduce the auxiliary notion of *factorization*.

Definition 1 (Factorization).
A factorization *of an element u of a semigroup $(U, \oplus)$ is a sequence $(u_1, \ldots, u_n)$, with $\forall i \in [\![1, n]\!], u_i \in U$, such that $u_1 \oplus \ldots \oplus u_n = u$.*

Each term u_i in a factorization of u is a *factor* of u. We note $f_u(i) = u_i$ and $|\, f_u \,| = n$. If $I = \{i_1 \ldots i_k\} \subseteq [\![1, n]\!]$, we also note $f_u(I) = u_{i_1} \oplus \ldots \oplus u_{i_k}$.

By taking into account the alternation constraint between the terms of a factorization, we can propose the following definition for analogical proportions.

Definition 2 (Analogical proportion (semigroups)).
Let $(x, y, z, t) \in U^4$, we say that $x : y :: z : t$ if and only if there exist some factorizations f_x, f_y, f_z, f_t, respectively of x, y, z and t, such that:

$$\forall i \in [\![1, d]\!], (f_y(i), f_z(i)) \in \{(f_x(i), f_t(i)), (f_t(i), f_x(i))\}.$$

By construction, we have $|\, f_x \,| = |\, f_y \,| = |\, f_z \,| = |\, f_z \,| = d$. The smallest integer d for which this property holds is termed the degree *of the proportion.*

Definition 2 expresses the alternation constraint by stipulating that all the factors $f_y(i)$ and $f_z(i)$ in y and z should be alternatively equal to factors in x or in t.

This definition is fairly general: it applies to any semigroup and *a fortiori* to any richer structure, such as those displayed in the inheritance graph of Figure 7. If the algebraic structure $(U, \oplus)$ has additional properties (commutativity, existence of a neutral element, existence of a unique inverse for $\oplus$), then this definition can be simplified. We study several cases of such simplifications in the following paragraphs.

Before turning our attention to these simplications, we introduce a first result, which will prove useful in the following.

Proposition 1.
Analogical proportions are preserved by semigroup morphisms,[7] *i.e. if $M : (U, \oplus) \to (V, \odot)$ is a (semigroup) morphism, then*

$$\forall (x, y, z, t) \in U^4, x : y :: z : t \Rightarrow M(x) : M(y) :: M(z) : M(t).$$

Proof. It suffices to note that if $f_u = (u_1, \ldots, u_n)$ is a factorization of $u \in (U, \oplus)$, then $F_u = (M(u_1), \ldots, M(u_n))$ is a factorization of $M(u) \in (V, \odot)$, since $M(u) = M(u_1 \oplus \ldots \oplus u_n) = M(u_1) \odot \ldots \odot M(u_n)$. □

[7] A (semigroup) morphism M from $(U, \oplus)$ to $(V, \odot)$ is a mapping such that $\forall x, y \in U, M(x \oplus y) = M(x) \odot M(y)$.

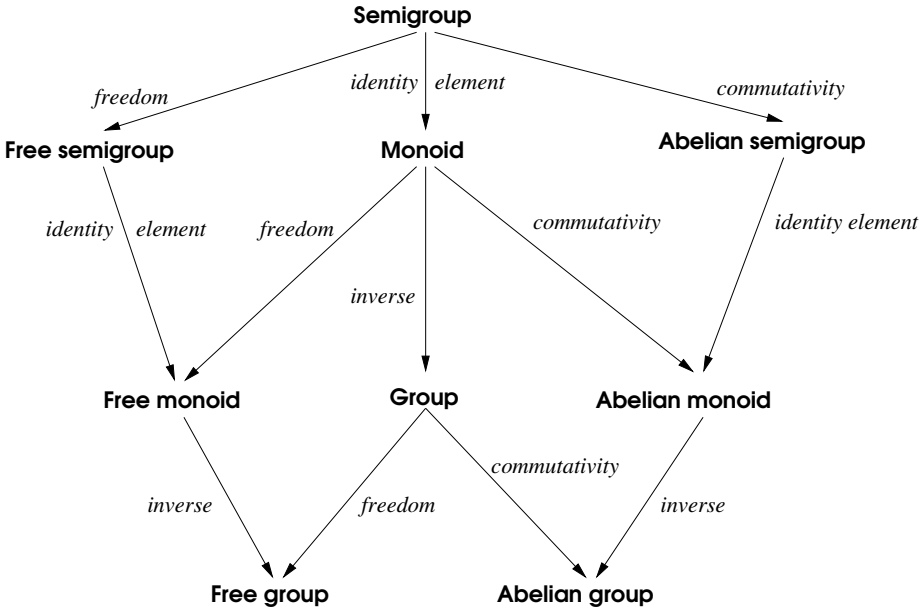


Figure 7: Inheritance graph of algebraic structures

3.2 Abelian Semigroups and Abelian Groups

When the internal composition law in a semigroup U is commutative, the semigroup is said to be commutative or *Abelian*. Within an Abelian semigroup, the order of factors in a factorization is not relevant. When analyzing a proportion, it is thus possible to reorder the terms in a factorization of y so that the factors shared with x are put together on the "left part" of the factorization; and the factors shared with t are on the "right part". The same applies for z, hence the following result.

Proposition 2.
If U is an Abelian semigroup, $\forall (x,y,z,t) \in U^4$, we have $x : y :: z : t$ if and only if either $(y,z) \in \{(x,t),(t,x)\}$ or $\exists (x_1,x_2,t_1,t_2) \in U^4$ such that $x = x_1 \oplus x_2,\ y = x_1 \oplus t_2, z = t_1 \oplus x_2,\ t = t_1 \oplus t_2$.

Proof. One implication is true by virtue of definition 2.

To prove the converse, let us assume that there exists a quadruple of factorizations $(f_x, f_y, f_z, f_t) \in (U^d)^4$ and two subsets I, J of $[\![1,d]\!]$ such that $I \cap J = \emptyset$, $I \cup J = [\![1,d]\!]$, and:

$$f_x(I) = f_y(I), f_z(I) = f_t(I), f_x(J) = f_z(J), f_y(J) = f_t(J).$$

The composition law $\oplus$ being commutative, $\forall \alpha \in \{x,y,z,t\}$, $(f_\alpha(I), f_\alpha(J))$ is a factorization of α, hence the result. □

An Abelian group is an Abelian semigroup, i.e. a commutative monoid with identity element 1_U in which each element u has a unique inverse $\ominus u$ such that $u \oplus \ominus u = 1_U$. The definition of analogical proportions in Abelian groups can be further simplified.

Proposition 3.
If U is an Abelian group, $\forall (x, y, z, t) \in U^4$, we have $x : y :: z : t$ if and only if

$$x \oplus (\ominus y) = z \oplus (\ominus t).$$

Proof. Proposition 2 readily yields:

$$x \oplus (\ominus y) = x_2 \oplus (\ominus t_2) = z \oplus (\ominus t).$$

Conversely, if $x \oplus (\ominus y) = z \oplus (\ominus t)$, taking:

$$x_1 = 1_U, x_2 = x, t_1 = z \oplus (\ominus x) \text{ and } t_2 = y,$$

yields:

$$x = x_1 \oplus x_2, y = x_1 \oplus t_2, z = t_1 \oplus x_2 \text{ and } t = t_1 \oplus t_2. \qquad \square$$

This definition is consistent with traditional and intuitive visions of analogical proportions. For example, in $(\mathbb{R}^\star, \times)$, it corresponds to the classical proportionality relation ($\frac{x}{y} = \frac{z}{t}$); in a vector space, it expresses the relation between the summits of a parallelogram ($\vec{x} - \vec{y} = \vec{z} - \vec{t}$, cf. Figure 8).

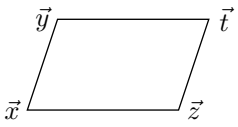


Figure 8: A parallelogram

From a computational perspective, these simplifications make the computational problems of validating and solving a proportion a trivial issue, which basically amounts to performing two operations in U.

As we shall see in the coming sections, the same definition 2 that is used here to account for proportions between the summits of a parallelogram, or between integers in $(\mathbb{N}, +)$, will be used to establish proportions between sequences of letters or between trees, which is a positive sign of the consistency of our model.

3.3 Sets and feature structures

The abstract algebraic framework that has been presented in the previous section provides us with a definition of analogical proportion which readily applies to sets, multisets, and feature structures.

3.3.1 Sets

The union operation is an internal operation defined on the powerset $(2^S, \cup)$ of a set S which makes it an Abelian monoid: it is associative, commutative and the empty set $\emptyset$ is the identity element. Proposition 2 directly applies.

Proposition 4.
Let S be a set, $\forall (x, y, z, t) \in (2^S)^4$, we have $x : y :: z : t$ if and only if $\exists (x_1, x_2, t_1, t_2) \in (2^S)^4$ such that:

$$x = x_1 \cup x_2,\ y = x_1 \cup t_2,\ z = t_1 \cup x_2,\ t = t_1 \cup t_2.$$

It is routine to check that this definition accounts for the example given in Section 2.3 (cf. Figure 3).

3.3.2 Multisets

A multiset can be seen as a set whose elements have a multiplicity. Formally, a multiset is a pair (S, m) where S is a set (called the underlying set of elements) and $m : S \to \mathbb{N}$ a function from S to the set $\mathbb{N}$ of (positive) natural numbers. For each element e in S, the multiplicity of e is $m(e)$.[8] For convenience, a multiset (A, m) can also be noted $\sum_{a \in A} m(a)a$; for example, the multiset $(\{a, b\}, m)$ such that $m(a) = 2$ and $m(b) = 1$ will be noted $2a + b$.

A multiset is an Abelian monoid with respect to the sum operation ($\biguplus$), defined as follows:

$$(A, m) \biguplus (B, n) = (A \cup B, f), \text{ where } f(e) = m(e) + n(e), \forall e \in A \cup B.$$

Therefore, we can again apply proposition 2. We can easily verify that the following proportion holds:

$$3A + r + s + l : m + u + r + s + 2i + l + n :: 3A + s + l + m : 2m + u + s + l + 2i + n,$$

which is also a consequence of proposition 1, since: (i) the proportion between words $ArsAlA : mursilin :: AslAmA : muslimin$ is verified (this will be formally proved in Section 3.4), (ii) the application that turns a word into a multiset of letters is a semigroup morphism.

3.3.3 Feature structures

A *feature structure* is made up of a set of attribute-value pairs. A value can be either atomic or complex (most commonly a feature structure[9]). Such representations are classical knowledge representation tools. Feature structures are also used in several formal models of syntax, such as LFG (*Lexical Functional Grammar*, [Bresnan, 2001]) and HPSG (*Head-Driven Phrase Structure Grammar*, [Pollard and Sag, 1994]); Carpenter [1992] gives an introduction to feature structures and their uses in NLP. A (matrix) representation of such a feature structure is displayed in Figure 9.

$$\begin{bmatrix} wordform : \textit{loves} \\ lemma : \texttt{love} \\ cat : \texttt{verb} \\ agreement : \begin{bmatrix} person : \texttt{third} \\ number : \texttt{singular} \end{bmatrix} \end{bmatrix}$$

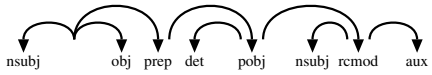
Figure 9: Feature structure representing the verb form *loves*

Feature structures can also be used to represent dependency relations, as produced by dependency parsers [Mel'čuk, 1988, Buchholz and Marsi, 2006], as displayed in Figure 10.

Formally, the set of feature structures FS is a set equipped with two binary internal composition laws which are associative, idempotent, and commutative. The internal composition laws of the set FS are the *unification* operation, denoted $\sqcup$ and the *generalization* operation, denoted $\sqcap$. The set FS is also equipped with a partial order

[8] For a multiset (S, m), we can also extend m to a set $U \supseteq S$, by setting $m(e) = 0, \forall e \in U \setminus S$.

[9] Some linguistic theories make also use of other types of complex feature values, such as lists or sets.



$$\begin{bmatrix} wordform : \text{“}loved\text{”} \\ nsubj. : \left[\, wordform : \text{“}she\text{”} \,\right] \\ prep. : \begin{bmatrix} wordform : \text{“}for\text{”} \\ pobj. : \begin{bmatrix} wordform : \text{“}dangers\text{”} \\ det. : \left[\, wordform : \text{“}the\text{”} \,\right] \\ rcmod. : \begin{bmatrix} wordform : \text{“}had\text{”} \\ nsubj. : \left[\, wordform : \text{“}I\text{”} \,\right] \\ aux. : \left[\, wordform : \text{“}passed\text{”} \,\right] \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ obj. \left[\, wordform : \text{“}me\text{”} \,\right] \end{bmatrix}$$

Figure 10: Feature structure representing the dependency graph of a sentence

relation, namely *subsumption*, denoted $\sqsubseteq$. We have $a \sqsubseteq b$ if a is more general than b, i.e. if $a \sqcap b = a$.

The set of feature structures being an Abelian semigroup for the unification operation, proposition 2 is also applicable.

Proposition 5.
$\forall (x, y, z, t) \in FS^4$, we have $x : y :: z : t$ if and only if $\exists (x_1, x_2, t_1, t_2) \in FS^4$ such that

$$x = x_1 \sqcup x_2,\ y = x_1 \sqcup t_2,\ z = t_1 \sqcup x_2,\ t = t_1 \sqcup t_2.$$

It has to be noted that this definition applies to any kind of feature structure, the only requirement being the definition of the unification operation. Once again, it can be checked that this definition covers the introductory examples of Section 2.4.

3.3.4 Lattices

In this section, we present an additional simplification of previous definitions, which allows us to get rid of the existential quantifiers when the underlying algebraic structure is a lattice. The two cases of data structures studied in this section, namely sets and feature structures, are particular cases of lattices and will thus benefit from this simplified definition.

A *lattice* is a mathematical structure which can be seen either as a partially ordered set (poset) or as an algebra. A lattice can be defined as a poset in which all nonempty finite subsets have a least upper bound (also called *supremum* or *join*) and a greatest lower bound (also called *infimum* or *meet*). An algebra $(L; \wedge, \vee)$ is called a lattice if L is a nonempty set, $\wedge$ and $\vee$ are binary operations in L, both idempotent, commutative, and associative, and satisfying the absorption law. These two views on lattices can easily be reconciled, and any lattice defined as an algebra can be turned into a lattice seen as a poset and vice-versa. This is stated in the following classical theorem [Grätzer, 1971]:

Theorem 1. *1. Let the poset $\mathcal{L} = (L; \leq)$ be a lattice. Define $a \wedge b = \inf\{a, b\}$ and $a \vee b = \sup\{a, b\}$. Then the algebra $\mathcal{L}^a = (L; \wedge, \vee)$ is a lattice.*

2. *Let the algebra $\mathcal{L} = (L; \wedge, \vee)$ be a lattice. Let $a \leq b$ if and only if $a \wedge b = a$ (or equivalently $a \vee b = b$). Then $\mathcal{L}^p = (L; \leq)$ is a poset, and the poset $\mathcal{L}^p$ is a lattice.*

3. *Let the poset $\mathcal{L} = (L; \leq)$ be a lattice; then $(\mathcal{L}^p)^a = \mathcal{L}$.*

4. *Let the algebra $\mathcal{L} = (L; \wedge, \vee)$ be a lattice; then $(\mathcal{L}^a)^p = \mathcal{L}$.*

The set of feature structures can be seen as a lattice with respect to the unification and generalization operations. The same applies to the powerset of a set with respect to the union and intersection operations. The following simplified definition of analogical proportion applies to these two types of representations.

Proposition 6.
If $(L; \wedge, \vee)$ is a lattice, $\forall (x, y, z, t) \in L^4$, we have $x : y :: z : t$ if and only if:

$$\begin{aligned} x &= (x \wedge y) \vee (x \wedge z), \\ y &= (x \wedge y) \vee (t \wedge y), \\ z &= (t \wedge z) \vee (x \wedge z), \\ t &= (t \wedge z) \vee (t \wedge y). \end{aligned}$$

Proof. $\Leftarrow$ Trivial as the formulation of proposition 6 is a particular case of the one used in proposition 2.

$\Rightarrow$ Let (x, y, z, t) be in L^4 such that $x : y :: z : t$. By definition, there exists some $(x_1, x_2, t_1, t_2) \in L^4$ such that

$$x = x_1 \vee x_2, y = x_1 \vee t_2, z = t_1 \vee x_2 \text{ and } t = t_1 \vee t_2.$$

We will first show that $x = (x \wedge y) \vee (x \wedge z)$. The inequalities

$$(x \wedge y) \vee (x \wedge z) \leq x \wedge (y \vee z) \leq x$$

hold from the basic properties of meet and join. If $a \leq b$ and $c \leq d$, then $a \leq b \leq (b \vee d)$ and $c \leq d \leq (b \vee d)$, yielding $a \vee c \leq b \vee d$. Since $x_1 \leq x$ and $x_1 \leq y$, we have $x_1 \leq (x \wedge y)$. Likewise, we have $x_2 \leq (x \wedge z)$. Consequently,

$$x = x_1 \vee x_2 \leq (x \wedge y) \vee (x \wedge z).$$

We therefore have

$$x = (x \wedge y) \vee (x \wedge z),$$

and the other equalities hold by symmetry. □

Based on this proposition, we can conclude that an algorithm for validating a proportion on sets or on feature structures will have the same complexity as the computation of the basic operations join and meet: in fact, validating an analogical proportion only involves 8 basic operations (4 meets and 4 joins). In the case of Boolean algebra, in which each element e has a complement $\neg e$ such that $e \wedge \neg e = 0$, $e \vee \neg e = 1$ and the two internal laws are mutually distributive, we have an additional result for the solving procedure.

Proposition 7.
If A is a Boolean algebra, we have $x : y :: z : t$ for $(x, y, z, t) \in A^4$ if and only if $x \leq (y \vee z)$ and $(y \vee z) \wedge \neg x \leq t \leq (y \vee z)$.

Proof. $\Rightarrow$. We assume that the proportion $x : y :: z : t$ holds. We thus have $t = (t \wedge z) \vee (t \wedge y)$. From the basic properties of meet and join, we deduce $t \leq (y \vee z) \wedge t \leq (y \vee z)$. Moreover, if we note $y_1 = (x \wedge y)$, $y_2 = (t \wedge y)$, $z_1 = (t \wedge z)$, and $z_2 = (x \wedge z)$, we have:

$$\begin{aligned}(y \vee z) \wedge \neg x =& ((y_1 \vee z_2) \vee (z_1 \vee y_2)) \wedge \neg(y_1 \vee z_2) \\ =& (z_1 \vee y_2) \wedge \neg(y_1 \vee z_2) \\ =& t \wedge \neg x \leq t,\end{aligned}$$

so $(y \vee z) \wedge \neg x \leq t \leq (y \vee z)$. Moreover, since $x = (x \wedge y) \vee (x \wedge z)$, we have $x \leq (y \vee z)$.

$\Leftarrow$. We assume $x \leq (y \vee z)$ and $(y \vee z) \wedge \neg x \leq t \leq (y \vee z)$. Since $x \leq (y \vee z)$, we have $x \leq x \wedge (y \vee z)$, and $x = (x \wedge y) \vee (x \wedge z)$. We have $(t \wedge z) \vee (t \wedge y) \leq t$ from the basic properties of meet and join. Moreover, since $t \leq (y \vee z)$, we have $t \leq t \wedge (y \vee z)$, so $t = (t \wedge z) \vee (t \wedge y)$. Since $(y \vee z) \wedge \neg x \leq t$, we have $((y \vee z) \wedge \neg x) \vee x \leq t \vee x$, and $((y \vee z) \vee x) \leq t \vee x$. Consequently, we have both $y \leq t \vee x$ and $z \leq t \vee x$, so $y \leq y \wedge (t \vee x)$ and $z \leq z \wedge (t \vee x)$. Finally, since $(x \wedge y) \vee (t \wedge y) \leq y$ and $z = (t \wedge z) \vee (x \wedge z) \leq z$ always hold, we can deduce $y = (x \wedge y) \vee (t \wedge y)$ and $z = (t \wedge z) \vee (x \wedge z)$. □

3.4 Words

This section is devoted to the study of the free monoid structure, which is the classical model to represent finite sequences of symbols, that is, finite words. We use definition 2, which can be directly applied to the case of words. In particular, we examine the verification of the solving procedures and present a framework based on finite-state transducers.

A detailed presentations of proportions on words, including proofs, can be found in [Yvon, 2003, Yvon et al., 2004, Stroppa, 2005], from where most definitions and constructions are borrowed. The discussion of the algorithmic aspects at the end of this section is entirely original.

3.4.1 Notations

Let Σ be a finite set of symbols, called an *alphabet*. $\Sigma^\star$ denotes the set of finite sequences of elements of Σ, called *words* over Σ. Provided with the *concatenation* operation, $\Sigma^\star$is a *free monoid* whose *identity element* is the *empty word* ϵ. For x and y in $\Sigma^\star$, the concatenation of x and y is denoted $x.y$ or simply xy. We note $|\,x\,|$ the *length* of x; we have $|\,\epsilon\,| = 0$. For $x \in \Sigma^\star$ and $i \leq |\,x\,|$, $x(i)$ denotes the i^{th}symbol in x.

3.4.2 Definition

For free monoids, we can rewrite definition 2; concatenation is the internal law used to define factorization in this case.

Definition 2 (bis).
If $\Sigma^\star$ is a free monoid, $\forall (x, y, z, t) \in {\Sigma^\star}^4$, we have $x : y :: z : t$ if and only if there exist some factorizations $(f_x, f_y, f_z, f_t) \in ((\Sigma^\star)^d)^4$ such that

$$\forall i \in [\![1, d]\!], (f_y(i), f_z(i)) \in \{(f_x(i), f_t(i)), (f_t(i), f_x(i))\}.$$

The smallest integer d for which this property holds is termed the degree *of the proportion.*

We can show [Yvon, 2003] that this definition generalizes the (algorithmic) definition of proportions between words proposed by Lepage [1998]; in both cases the following properties hold. First, an analogical equation may have one, zero, or multiple solutions. Lepage [2001] gives a number of conditions for an equation to have at least one solution; these conditions also apply here. In particular, if t is a solution of $x : y :: z :?$, then t contains all the symbols in y and z that are not in x, in the same order. As a corollary, all the solutions of an analogical equation (between words) have the same length.[10]

3.4.3 A Finite-state Solver

Proposition 1 yields an efficient procedure for solving analogical equations, based on finite-state transducers. We only sketch here the main steps of the procedure and we refer the reader to the above cited papers for a more detailed account. To start with, let us introduce the notions of *complementary set* and *shuffle product*.

Complementary set If v is a subword of w, the *complementary set* of v with respect to w, denoted by $w\backslash v$ is the set of subwords of w obtained by removing from w, from left to right, the symbols in v. For instance, *eea* belongs to the complementary set of *xmplr* with respect to *exemplar*. When v is not a subword of w, $w\backslash v$ is empty. This notion can be generalized to any regular language.

The complementary set of v with respect to w is a regular set: it is the output language of the finite-state transducer T_w (see Figure 11) for the input v.

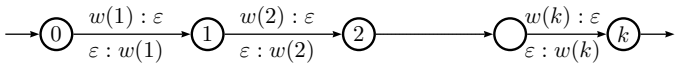


Figure 11: The transducer T_w computing complementary sets with respect to w.

Shuffle The *shuffle* $u \bullet v$ of two words u and v is defined as follows [Sakarovitch, 2003]:

$$u \bullet v = \{u_1 v_1 u_2 v_2 \dots u_n v_n, \text{ with } u_i, v_i \in \Sigma^\star, u_1 \dots u_n = u, v_1 \dots v_n = v\}.$$

The shuffle of two words u and v contains all the words w which can be composed using all the symbols in u and v, subject to the condition that if a precedes b in u (or in v), then it precedes b in w. Taking, for instance, $u = abc$ and $v = def$, the words $abcdef$, $abdefc$, $adbecf$ all belong to $u \bullet v$; this is not the case of $abefcd$. This operation generalizes straightforwardly to languages.

The shuffle of two regular languages is regular [Sakarovitch, 2003]; the automaton A, computing $K \bullet L$, is derived from the automata $A_K = (\Sigma, Q_K, q_K^0, F_K, \delta_K)$ and $A_L = (\Sigma, Q_L, q_L^0, F_L, \delta_L)$ recognizing respectively K and L as the product automata $A = (\Sigma, Q_K \times Q_L, (q_K^0, q_L^0), F_K \times F_L, \delta)$, where δ is defined as: $\delta((q_K, q_L), a) = (r_K, r_L)$ if and only if either $\delta_K(q_K, a) = r_K$ and $q_L = r_L$ or $\delta_L(q_L, a) = r_L$ and $q_K = r_K$.

[10] Note that this result is directly derived from proposition 1, since the length application is a semigroup morphism from the set of words $\Sigma^\star$ to the set of integers $\mathbb{N}$.

The notions of complementary set and shuffle are related through the following property, which is a direct consequence of the definitions.

$$\forall w, u, v \in \Sigma^\star,\ \ w \in u \bullet v \Leftrightarrow u \in w \backslash v.$$

Solving analogical equations The notions of shuffle and complementary sets yield another characterization of analogical proportion between words, based on the following proposition [Yvon, 2003]:

Proposition 8.

$$\forall x, y, z, t \in \Sigma^\star,\ x : y :: z : t \Leftrightarrow x \bullet t \cap y \bullet z \neq \emptyset.$$

An analogical proportion is thus established if the symbols in x and t are also found in y and z, and appear in the same relative order. A corollary follows:

Proposition 9.

$$t \textit{ is a solution of } x : y :: z :? \Leftrightarrow t \in (y \bullet z) \backslash x.$$

The set of solutions of an analogical equation $x : y :: z :?$ is a regular set, which can be computed with a finite-state transducer. It can also be shown Yvon et al. [2004] that this analogical solver generalizes the approach based on edit distance proposed in Lepage [1998].

3.4.4 Computational issues

Given the finite-state constructions presented above, it is possible to assess the complexity of the validation and solving procedures. In the case of validation, we have to build the shuffles $x \bullet t$ and $y \bullet z$, which yield two automata of size $|x+1| \times |t+1|$ and $|y+1| \times |z+1|$ respectively. We then have to intersect these automata, yielding an automaton of a most $|x+1| \times |y+1| \times |z+1| \times |t+1|$, which is then tested for emptiness.[11]

The solving procedure is similar: we need to build the shuffle automaton of y and z, then to compute its complementary set with respect to x. Consequently, there is a non-deterministic automaton which represents the set of all possible solutions of $x : y :: z :?$ with $O(|x| \times |y| \times |z|)$ states. Enumerating all the words in this non-deterministic automaton is an exponential process, though this limit is rarely met in practice.

In the following, we present several ways to associate a numerical score to a proportion; these scores can be used to reduce the complexity of the solving procedure.

3.4.5 Scoring analogical proportions

Scoring analogical proportions serves two purposes. First, as analogical equations may have more than one solution, we need to define criteria allowing the comparison and ranking of competing solutions. Second, scoring proportions enables us to consider only a subset of optimal solutions when solving an analogical equation, which may reduce the complexity of the procedure. In the following, we present several score functions and discuss some of their properties. Additional measures of analogical proportions are presented in [Yvon et al., 2004], based on the aggregation of measures of proportions between symbols of Σ.

[11] It is possible to build the intersection product of two automata in such a way that all the states in the product are accessible [Sakarovitch, 2003]. In this case, checking for emptiness simply amounts to verifying

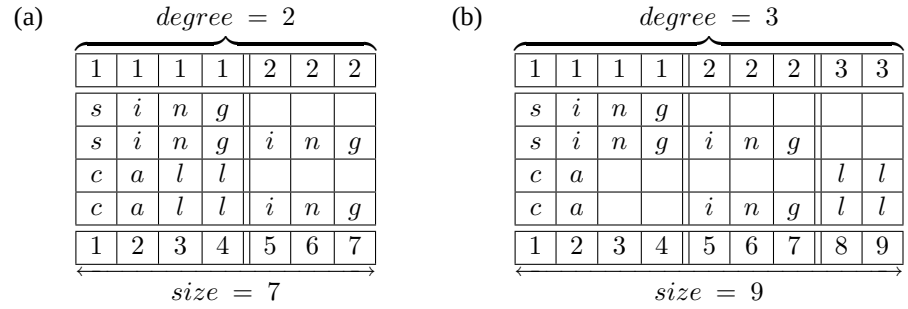


Figure 12: The degree and size of two proportions on words (empty cells correspond to empty factors (ϵ))

Degree The *degree* is a score function for proportions introduced in definition 2. The degree counts the number of alternating chunks in an analogical proportion: the smaller the degree, the simpler the proportion. The "trivial" proportions of the form $a : a :: b : b$ or $a : b :: a : b$ have a degree of 1. The degree is related to the number of factors found in the words involved in a proportion; a small degree corresponds the intuition that good proportions should preserve larger parts of the original words.

The degree of an analogical proportion $x : y :: z : t$ is denoted $D(x : y :: z : t)$ and can be alternatively defined as followed, where $\mathcal{F}$ denotes the set of factorizations for which the proportion $x : y :: z : t$ holds.

Definition 3 (Degree).
The degree *of an analogical proportion $x : y :: z : t$ is defined as follows:*

$$D(x : y :: z : t) = \min_{(f_x, f_y, f_z, f_t) \in \mathcal{F}} |\, f_x \,|$$

Size The degree counts the number of alternating factors. The *size* of a proportion counts the number of atomic symbol alignments, which take the form $a : a :: b : b$ or $a : b :: a : b$ (cf. Figure 12).

Definition 4 (Size).
The size *of an analogical proportion $x : y :: z : t$ is defined as:*

$$S(x : y :: z : t) = \min_{(f_x, f_y, f_z, f_t) \in \mathcal{F}} \sum_{i \in [\![1, |\, f_x \,|]\!]} \max(|\, f_x(i) \,|, |\, f_t(i) \,|)$$

To illustrate the notions of degree and size, two examples are displayed in Figure 12. Whereas the analogical proportion $sing : singing :: call : calling$ has a degree of 2 and a size of 7, $sing : singing :: call : caingll$ has a degree of 3 and a size of 9, in agreement with the intuition that the former proportion is better than the latter.

ϵ-order The ϵ-order of an analogical proportion is the maximum number of consecutive empty factors appearing in the factorizations involved in the proportion. For

that the set of final states is not empty, which is done in constant time.

example, the proportion $sing : singing :: call : calling$ has an ϵ-order of 1, whereas $sing : singing :: call : caingll$ has an ϵ-order of 2: two consecutive empty factors appear in the related factorization of $call$ (cf. Figure 12).

Definition 5 (ϵ-order).
The ϵ-order *of an analogical proportion* $x : y :: z : t$ *with respect to* $\alpha \in \{x, y, z, t\}$ *is defined as:*

$$O_\epsilon(x : y :: z : t,\ \alpha) = \min_{(f_x, f_y, f_z, f_t) \in \mathcal{F}} \max_{1 \leq i \leq j \leq |f_x|} \{|\, j - i + 1 \,| : f_\alpha([\![i, j]\!]) = \epsilon\}$$

The ϵ*-order for the whole proportion is defined as:*

$$O_\epsilon(x : y :: z : t) = \max_{\alpha \in \{x,y,z,t\}} \{O_\epsilon(x : y :: z : t,\ \alpha)\}$$

The ϵ-order is actually related to the notions of degree and size, yielding the following result.

Proposition 10.
If t *is a solution of the equation* $x : y :: z :?$ *such that*

$$D(x : y :: z : t) > O_\epsilon(x : y :: z : t,\ x) > 1,$$

then there exists a solution $\hat{t}$ *such that the following (in)equalities are satisfied:*

$$\begin{aligned} O_\epsilon(x : y :: z : \hat{t},\ x) &= O_\epsilon(x : y :: z : t,\ x) - 1, \\ D(x : y :: z : \hat{t}) &= D(x : y :: z : t) - 1, \\ S(x : y :: z : \hat{t}) &\leq S(x : y :: z : t). \end{aligned}$$

The proof of this proposition is given in Appendix A.

Corollary 1. *If* t *is a solution of the equation* $x : y :: z :?$ *with*

$$D(x : y :: z : t) \ > \ O_\epsilon(x : y :: z : t,\ x) \ > \ 1,$$

then there exists a solution $\hat{t}$ *such that*

$$O_\epsilon(x : y :: z : \hat{t},\ x) = 1.$$

In particular, this result implies that the search for a minimal degree solution of the equation $x : y :: z :?$ can be pruned by searching only those words t which satisfy the inequality

$$O_\epsilon(x : y :: z : t,\ x) \leq 1.$$

Reducing the complexity of the solving procedure Since the degree of an analogical proportion is a sign of its quality, it is possible to consider only solutions with a degree less than a given threshold.

To assess the complexity of this heuristic, let us first consider the case where the maximum degree is 2. In this case, factorizing three words amounts to considering a triple of positions which denotes the places where the factors separate. In Example (a) of Figure 12, the triple of positions of the factorizations is $(4, 4, 4)$ in (x, y, z); in Example (b), the degree of the proportion is 3, so there are two triples of positions:

$(4,4,2)$ and $(4,7,2)$. Once a triple of positions is chosen, we need to verify that the factor x_i is equal to y_i or z_i. This verification requires at most $|y| + |z| = m$ operations. Since there are $(|x|+1)\times(|y|+1)\times(|z|+1) = n$ triples of positions, the overall complexity is $n \times m$. If the maximal degree is d, then we have to consider $d-1$ triples of positions. The number of such triples is given by the multinomial coefficients. For a degree d, the number of factorizations of x is $(x, d-1)! = \binom{|x|+d-1}{d-1}$ (and likewise for y and z); the number of triples of positions is thus:

$$\binom{|x|+d-1}{d-1} \times \binom{|y|+d-1}{d-1} \times \binom{|z|+d-1}{d-1},$$

which is bounded by a polynomial of degree $d-1$, hence the global complexity:

$$O((|x| \times |y| \times |z|)^{d-1} \times m).$$

Here again, this theoretical complexity is rarely met in practice. A complementary approach consists in exploiting the result of the corollary 1. More precisely, if we are interested in solutions of low degree, then solutions involving consecutive epsilons may be safely skipped, which speeds up the search, though formal complexity is still polynomial in the product of the words lengths.

4 Multi-level proportions and proportions between languages

In this section, we present a generalization of the framework developed in the previous sections introduced in [Yvon et al., 2004]. This generalization is needed to define proportions over languages. We then propose two possible definitions and show that they are in fact equivalent.

In the algebraic framework presented above, the notion of analogical proportion relies on: (i) the ability to decompose complex entities into smaller parts, (ii) the existence of *alignments* between these parts. Recall that four objects (x, y, z, t) form an analogical proportion if and only if there exist some factorizations (f_x, f_y, f_z, f_t) of (x, y, z, t) such that:

$$\forall i \in [\![1, n]\!],\ (f_y(i), f_z(i)) \in \{(f_x(i), f_t(i)), (f_t(i), f_x(i))\}.$$

This latter condition states that it is possible to *align* the factors involved in the factorizations, forming quadruplets $(f_x(i), f_y(i), f_z(i), f_t(i)$ that either match the pattern $u : u :: v : v$ or $u : v :: v : u$. We have in fact already used this property when defining the size and the ϵ-order of a proportion between words over some finite alphabet. The left part of Figure 12 displays an alignment which supports the proportion $sing : singing :: call : calling$.

This property warrants the following alternative definition of proportions:

Definition 6 (Recursive formulation of analogical proportions).
For $(x, y, z, t) \in U^4$, we have $x : y :: z : t$ if and only if either $(y, z) \in \{(x, t), (t, x)\}$, or $\forall \alpha \in \{x, y, z, t\}$, $\exists(\alpha_1, \alpha_2) \in (U, \oplus)^2$ such that $\alpha = \alpha_1 \oplus \alpha_2$ and $\forall i \in [\![1, 2]\!]$, $x_i : y_i :: z_i : t_i$.

It can be proved [Stroppa, 2005] that definition 6 is equivalent to definition 2.

This new definition makes explicit the fact that analogies on composite objects (i.e. objects that decompose in factors) are based on simple alternations between their factors, which will be termed *base* proportions in the remainder of this section.

In light of this reformulation of proportions, a natural extension is the consideration of a richer set of base proportions, giving rise to the concept of *multi-level proportions*. This extension is explored in the following section.

4.1 Multi-level proportions

Allowing arbitrary base proportions, definition 2 is generalized as follows.

Definition 7 (Multi-level proportions).
Let $\mathcal{B}(U)$ denotes a set of base proportions*, i.e. a subset of U^4. For all $(x, y, z, t) \in U^4$, we have $x : y :: z : t$ if and only if either $(x, y, z, t) \in \mathcal{B}(U)^4$ or $\forall \alpha \in \{x, y, z, t\}$, $\exists (\alpha_1, \alpha_2) \in (U, \oplus)^2$ such that $\alpha = \alpha_1 \oplus \alpha_2$ and $\forall i \in [\![1, 2]\!]$, $x_i : y_i :: z_i : t_i$,*

Denoting $\mathcal{A}(U)$ the set of atomic proportions, definition 2 is obviously a particular case of definition 7 where $\mathcal{A}(U) = \mathcal{B}(U)$. However, $\mathcal{B}(U)$ may now contain more than the atomic proportions.

To illustrate this mechanism, let us consider the following example of analogical proportion between sequences of phonemes:

/ædvaɪs/ː/ædvaɪz/ːː/bɪliːf/ː/bɪliːv/ (*advice* : *advise* :: *belief* : *believe*)

Definition 1 does not recognize this quadruplet as forming a valid proportion: alternating prefixes are easily identified, but no quadruplet of alternating suffix exists, as the alignment (/s/ː/z/ːː/f/ː/v/) does not belong to the set of legitimate atomic proportions between individual phonemes. However, if we take into account the internal structural of the phoneme set, the proportion can be established. Let us consider each phoneme as a vector of distinctive binary features, as in Figure 4. In this context, the proportion (/s/ː/z/ːː/f/ː/v/) is valid, and represent an instance of the opposition between voiced and unvoiced consonants.

To sum up, by considering an extended notion of analogical proportion between elements of the generator set Σ of the free monoid, we are in a position to generalize the notion of analogical proportion between elements of $\Sigma^\star$. This strategy is fairly general, and allows us to stack together various levels of analogical proportions: it allows us, for instance, to define proportions between strings of sets, but also strings of strings, sets of features-structures, sets of strings of strings, etc.

This generalization has obvious applications in the modeling of language, where the atoms of syntax (be they word forms or morphemes) are themselves made up of combinatorial arrangements of phonological atoms, a situation sometimes referred to in linguistic circles as the *double articulation*. Using two levels of proportions, the following four sentences, viewed as sequences of words, do constitute a valid multi-level proportion:

dogs are resting by the fire	:	*the cat sleeps on the coach*
	::	
cats are sleeping on the coach	:	*the dog rests by the fire*

Figure 13: A two-level proportion

In the following, we study the case where a definition of analogical proportions is available for a part $P \subseteq U$ (typically for elements in the generative set of U, when U is

a free semi-group). A proportion with respect to P between four entities $(x, y, z, t) \in P^4$ is denoted $x : y \overset{P}{::} z : t$; $\mathcal{B}(U)$ is then simply defined as

$$\mathcal{B}(U) = \{(x, y, z, t) \in P^4 | \, x : y \overset{P}{::} z : t\}.$$

This formal extension is sufficient to define proportions for sets of strings, that is for *languages*, to which we now turn our attention.

4.2 Languages

Languages are commonly encountered in NLP applications: finite languages, for instance, provide a convenient model for ambiguities. They can be used to represent alternative analyses of a word, pronunciation variants, or a set of possible translations of a sentence.

An example of a proportion between finite languages, drawn from the word pronunciation task, is reproduced in Figure 14. As this example makes clear, our prime

(potato) [pəˈtɑːtə] : [pəˈtɑːtə], [pəˈteɪtə]
::
(tomato) [təˈmɑːtə] : [təˈmɑːtə], [təˈmeɪtə]

Figure 14: Analogical proportion between languages.

interest in defining proportions between languages is to capture systematic ambiguities: in our example, the fact that a particular sequence of graphemes (ato) in a specific set of words is the subject of dialectal variations; in the context of automatic translation, the fact that two synonyms can be interchanged in almost every context.

A (finite) language over Σ is a (finite) subset of $\Sigma^\star$. The set of languages is a semigroup with respect to the classical union operation. It is also a semigroup with respect to the concatenation operation, defined as:

$$L.K = \{lk \mid l \in L, k \in K\}.$$

Moreover, the concatenation is distributive with respect to the union and the neutral element for the union is a zero for concatenation, which makes the set of languages a semiring.

Languages can thus be studied from two different perspectives: (i) as sets of words and (ii) as elements of a semiring. Each of these perspectives entails a definition of analogical proportions between languages: the former suggests the use of the multi-level framework presented above; the latter points to an adaptation of definition 2 to algebraic structures equipped with two operators. In the following, we study these two definitions in more depth. They will eventually prove to be equivalent, which is a new piece of evidence in favour of the consistency of our framework.

4.2.1 Languages as sets of words

The set of languages over an alphabet Σ is $2^{\Sigma^\star}$. A language being a set of words, the multi-level framework is applicable: $\mathcal{B}(2^{\Sigma^\star})$ is based on the definition of (usual or extended) analogical proportions between words, which correspond to the singletons in $2^{\Sigma^\star}$.

In order to distinguish between words and languages, analogical proportions between words will be denoted by $x : y \overset{\Sigma^\star}{::} z : t$ in the following. For example, since $reading : reader \overset{\Sigma^\star}{::} reviewing : reviewer$, we have (over $2^{\Sigma^\star}$):

$$\{reading\} : \{reader\} :: \{reviewing\} : \{reviewer\}.$$

This is formally expressed by:[12]

$$\mathcal{B}(2^{\Sigma^\star}) = \{(\{x\},\{y\},\{z\},\{t\}) \in \{\{w\}|w \in \Sigma^\star\}^4 |\ x : y \overset{\Sigma^\star}{::} z : t\} \cup \mathcal{A}(2^{\Sigma^\star}).$$

In this context, the definition of analogical proportion between words acts as a bootstrap for the case of languages. Our introductory example is correctly covered by this definition, and it is routine to check that:

$$\begin{aligned}
\{\ \text{pəˈtɑːtə}\} &= \{\ \text{pəˈtɑːtə}\} \cup \{\ \text{pəˈtɑːtə}\} \\
\{\ \text{pəˈtɑːtə, pəˈteɪtə}\} &= \{\ \text{pəˈtɑːtə}\} \cup \{\ \text{pəˈteɪtə}\} \\
\{\ \text{təˈmɑːtə}\} &= \{\ \text{təˈmɑːtə}\} \cup \{\ \text{təˈmɑːtə}\} \\
\{\ \text{təˈmɑːtə, təˈmeɪtə}\} &= \{\ \text{təˈmɑːtə}\} \cup \{\ \text{təˈmeɪtə}\}
\end{aligned}$$

with

$$\text{pəˈtɑːtə} : \text{pəˈtɑːtə} \overset{\Sigma^\star}{::} \text{təˈmɑːtə} : \text{təˈmɑːtə}$$

and

$$\text{pəˈtɑːtə} : \text{pəˈteɪtə} \overset{\Sigma^\star}{::} \text{təˈmɑːtə} : \text{təˈmeɪtə}.$$

This notion of analogical proportion between languages will be called *WLang* and denoted $x : y \overset{W}{::} z : t$. The degree for a *WLang* proportion is defined as for the case of semigroups.

4.2.2 Languages as elements of a semiring

As already noted in Section 4.2, the set of languages is a semigroup with respect to both the classical union operation and the concatenation operation, which makes it a *semiring*. Building on this dual nature, it is possible to propose another generalization of analogical proportions, readily applicable to the case of languages. This generalization extends definition 2 as follows.

Definition 8 (Analogical proportion (in a semiring)).
If $(U, \oplus, \otimes)$ is a semiring, $\forall (x,y,z,t) \in U^4$, we have $x : y :: z : t$ if and only if either $(y,z) \in \{(x,t),(t,x)\}$, or $\forall \alpha \in \{x,y,z,t\}$, $\exists (\alpha_1, \alpha_2) \in U^2$ and $\odot \in \{\oplus, \otimes\}$ such that $\alpha = \alpha_1 \oplus \alpha_2$ and $\forall i \in [\![1,2]\!]$, $x_i : y_i :: z_i : t_i$.[13]

This definition is a simple extension of definition 2, in which two operators, rather than just one, can be used in factorizations.

A proportion on languages can thus be defined as a specific instance of definition 8, where the union and concatenation operations respectively instantiate $\oplus$ and $\otimes$. In this context, the notion of degree is well-defined and continues to refer to one plus the minimal number of recursions involved in a proportion.

This definition also correctly captures our working example, as:

[12] Since atomic proportions have served as a basic component of our formalism, it seems reasonable to impose $\mathcal{A}(U) \subseteq \mathcal{B}(U)$; this is the reason why $\mathcal{A}(2^{\Sigma^\star})$ is explicitly added to $\mathcal{B}(2^{\Sigma^\star})$.

[13] Since the multiplication ($\otimes$) with the identity element for $\oplus$ is an absorbing (thus destructive) operation, we additionally impose that this identity element cannot be involved in a multiplication.

$$
\begin{aligned}
\{\ \text{pəˈtɑːtə}\} &= \{\ \text{pəˈt}\}\ .\ \{\ \text{ɑː}\}\ .\ \{\ \text{tə}\} \\
\{\ \text{pəˈtɑːtə, pəˈteɪtə}\} &= \{\ \text{pəˈt}\}\ .\ \{\ \text{ɑː, eɪ}\}\ .\ \{\ \text{tə}\} \\
\{\ \text{təˈmɑːtə}\} &= \{\ \text{təˈm}\}\ .\ \{\ \text{ɑː}\}\ .\ \{\ \text{tə}\} \\
\{\ \text{təˈmɑːtə, təˈmeɪtə}\} &= \{\ \text{təˈm}\}\ .\ \{\ \text{ɑː, eɪ}\}\ .\ \{\ \text{tə}\}.
\end{aligned}
$$

This notion of analogical proportion between languages will be called *SLang* and denoted $x : y \overset{S}{::} z : t$. In the case of *SLang*, the notion of degree is defined as the number of recursions involved in the proportion.

4.2.3 An equivalence result

In this section, we prove that the two definitions considered above are actually equivalent in the case of finite languages. Before developing the equivalence result, let us informally compare these definitions. Both involve two components: a definition of base proportions and a recursive mechanism of composition. In *WLang*, base proportions include atomic proportions and proportions between isolated words, and aggregation is made using the union operation; in contrast, in *SLang*, base proportions are the only atomic ones, but they can be recombined using both union and concatenation. *WLang* is based on an enriched notion of base proportion, while *SLang* draws upon a more liberal aggregation procedure. The proof additionally needs the following lemma.

Lemma 1. *An atomic proportion between non-empty finite languages can be reinterpreted in terms of factorizations which only involve singletons.*

Proof. Indeed, if $A : A \overset{W}{::} B : B$ with $A = \cup_{k=1}^{i}\{a_k\}$ and $B = \cup_{k=1}^{j}\{b_k\}$, we can write:

$$m = \max(i, j), a_k = a_1 \text{ for } k \in [\![i+1, m]\!] \text{ and } b_k = b_1 \text{ for } k \in [\![j+1, m]\!],$$

which yields: $\forall k \in [\![1, m]\!]$, $\{a_i\} : \{a_i\} \overset{W}{::} \{b_i\} : \{b_i\}$. □

Proposition 11.
$\forall (x, y, z, t) \in (2^{\Sigma^\star})^4$, $x : y \overset{W}{::} z : t \Leftrightarrow x : y \overset{S}{::} z : t$.

Proof. $\Rightarrow$. By induction on the degree of the proportion. Let $(x, y, z, t) \in (2^{\Sigma^\star})^4$ with $x : y \overset{W}{::} z : t$ and $d = D(x : y \overset{W}{::} z : t)$ the degree of the proportion. By definition, there exist some languages:

$$(x_i)_{i\in[\![1,d]\!]}, (y_i)_{i\in[\![1,d]\!]}, (z_i)_{i\in[\![1,d]\!]}, (t_i)_{i\in[\![1,d]\!]}$$

so that $\forall \alpha \in \{x, y, z, t\}$, $\alpha_1 \cup \cdots \cup \alpha_d = \alpha$, and $\forall i \in [\![1, d]\!]$, $x_i : y_i \overset{W}{::} z_i : t_i$ (atomic proportion or proportion between singletons). If $d = 1$, then the quadruplet (x, y, z, t) is either an atomic proportion or a proportion between singletons. In the former case, the result is trivially verified. In the latter, x, y, z and t being singletons, we can note $(x, y, z, t) = (\{x'\}, \{y'\}, \{z'\}, \{t'\})$ with $x' : y' \overset{\Sigma^\star}{::} z' : t'$. The definition of analogical proportion between words states that for $\alpha' \in \{x', y', z', t'\}$, there exist some factors $(\alpha'_i)_{i\in[\![1,d]\!]} \in \Sigma^\star$ so that $\alpha' = \alpha'_1 \ldots \alpha'_m$ and

$$\forall i, (y'_i, z'_i) \in \{(x'_i, t'_i), (t'_i, x'_i)\}.$$

If we set $\alpha_i = \{\alpha'_i\}$, we then have: $\alpha = \alpha_1 \ldots \alpha_m$ (by concatenation of languages) with $\forall i \in [\![1, m]\!]$, $(y_i, z_i) \in \{(x_i, t_i), (t_i, x_i)\}$ and $\alpha_i \neq \emptyset$. Since the definition of

SLang allows for concatenations, we have $x : y :: z : t$, which establishes the implication for $d = 1$. The remaining of the inductive proof is trivial, as the composition mechanism is richer in the case of *SLang* than it is for *WLang*.

$\Leftarrow$. By induction on the degree of the proportion. Let $(x, y, z, t) \in (2^{\Sigma^\star})^4$ with $x : y \overset{S}{::} z : t$ and $d = D(x : y \overset{S}{::} z : t)$ the degree of the proportion. If $d = 1$, then the result is trivial since $\mathcal{A}(2^{\Sigma^\star}) \subseteq \mathcal{B}(2^{\Sigma^\star})$. Let us now assume that the result holds for any proportion with degree $k < d$. By definition, $\exists x_1, x_2, t_1, t_2 \in (2^{\Sigma^\star})^4$ such that:

$$x = x_1 \odot x_2, y = x_1 \odot t_2, z = t_1 \odot x_2, t = t_1 \odot t_2,$$

with $\odot \in \{\cup, .\}$, $x_1 : y_1 \overset{S}{::} z_1 : t_1$ and $x_2 : y_2 \overset{S}{::} z_2 : t_2$. By definition of the degree of a proportion, we have $D(x_1 : y_1 \overset{S}{::} z_1 : t_1) < d$ and $D(x_2 : y_2 \overset{S}{::} z_2 : t_2) < d$. Using the induction hypothesis, we thus also have $x_1 : y_1 \overset{W}{::} z_1 : t_1$ and $x_2 : y_2 \overset{W}{::} z_2 : t_2$. If $\odot = \cup$, the implication holds, since the union operation defines the aggregation mechanism of *WLang*, i.e. since:

$$x_1 : y_1 \overset{W}{::} z_1 : t_1 \wedge x_2 : y_2 \overset{W}{::} z_2 : t_2 \Rightarrow x : y \overset{W}{::} z : t.$$

If $\odot = .$, we can express the various factors involved as unions of proportions between singletons using Lemma 1,[14] i.e. $\forall \alpha \in \{x, y, z, t\}$, $\alpha_1 = \cup_{k=1}^{i} \{\alpha'_{1k}\}$, $\alpha_2 = \cup_{k=1}^{j} \{\alpha'_{2k}\}$ with:

$$\forall k \in [\![1, i]\!], x'_{1k} : y'_{1k} \overset{\Sigma^\star}{::} z'_{1k} : t'_{1k} \text{ and } \forall l \in [\![1, j]\!], x'_{2l} : y'_{2l} \overset{\Sigma^\star}{::} z'_{2l} : t'_{2l}.$$

It is consequently possible to write:

$$\forall \alpha \in \{x, y, z, t\}, \alpha = \alpha_1 \alpha_2 = \cup_{k=1,l=1}^{i,j} \{\alpha'_{1k}\}\{\alpha'_{2l}\} = \cup_{k=1,l=1}^{i,j} \{\alpha'_{1k}\alpha'_{2l}\}.$$

Since $\forall k \in [\![1, i]\!], x'_{1k} : y'_{1k} \overset{\Sigma^\star}{::} z'_{1k} : t'_{1k}$ and $\forall k \in [\![1, j]\!], x'_{2k} : y'_{2k} \overset{\Sigma^\star}{::} z'_{2k} : t'_{2k}$, we also have $\forall (k, l) \in [\![1, i]\!] \times [\![1, j]\!]$, $x'_{1k}x'_{2l} : y'_{1k}y'_{2l} \overset{\Sigma^\star}{::} z'_{1k}z'_{2l} : t'_{1k}$, which completes the proof that the proportion $x : y \overset{W}{::} z : t$ holds. □

4.2.4 Discussion

The equivalence result established above is a sign of the consistency of the framework. For finite languages, the verification and solving procedures are decidable, though no efficient algorithm is known. In this case, a brute-force approach remains possible, by considering $WLang$ and the set of factorizations (with respect to the union operation) of a given language.[15] We do not have such a result in the case of infinite languages.

[14] This is legitimate by virtue of the fact, as pointed out in Section 4.2.2, that the factors composed using concatenation are non-empty.

[15] The number of factorizations (with respect to union) of a given language is actually infinite since a factor can be repeated an unlimited number of times. However, if we have proportions in mind, it is clear that it is not necessary to repeat a term more than twice: in a proportion $x : y :: z : t$, a factor in x can only be aligned with a factor in y, in z, or in both. For a given language, the number of factorizations that are relevant to proportions is thus finite.

5 Tree proportions

Labeled trees are commonly used in Natural Language Processing to represent linguistic entities: for instance, they can represent syntactic structures, or terms in the logical representation of a concept or of a sentence. Hence the need to provide a sound definition, and companion algorithms, for computing proportions on trees.

The definition of proportions between trees is quite similar to the one used for words and involves (i) associative binary operations between trees and (ii) the notion of alternating subtrees. The most obvious candidate operation, which is commonly encountered in tree-based grammar formalisms such as TSG [Bod, 1992], is the left-most (label) substitution. However, this operation is not associative, which led us to ground our definition on the related notion of v-*substitution* (or substitution for short), which is the operation used to combine subtrees. This operation is introduced in section 5.1, followed in section 5.2 by a definition of tree proportions. We then turn to algorithmic issues, and successively study the case of exact and approximate computation of tree proportions. The main results of these studies are (i) exact algorithms for validating proportions between trees and for computing the solution of a proportion (section 5.3) and (ii) approximate algorithms for solving proportional equations on trees, which are based on more restrictive definition of proportions and exploit various linearizations of the input tree.

5.1 Notations and definitions

The following notations and definitions are mainly borrowed and/or adapted from Comon et al. [1997] and Shieber [2004].

Definition 9 (Trees).
Let L be a finite set of labels, *V a set of* variables, *and $U = L \cup V$. The set of trees over U, denoted $\mathcal{T}(U)$, is the smallest set such that:*

- $l \in \mathcal{T}(U)$ *for all* $l \in U$*;*
- $f(t_1, \ldots, t_n) \in \mathcal{T}(U)$ *for all* $f \in U$ *and* $i \in [\![1, n]\!]$*, with* $t_i \in \mathcal{T}(U)$.

Nodes in a tree are identified by their position, where a position is simply a sequence of positive integers, i.e. a word in $\mathbb{N}_+^\star$. Position ϵ is the position of the root, 1 the position of its first (leftmost) child, 1.2 the position of the second child of its first child, and so forth. A set $Pos(t)$ of nodes positions is *prefixial* if

$$\forall (u, i) \in (\mathbb{N}_+^\star \times \mathbb{N}_+), u.i \in Pos(t) \Rightarrow \forall j \in [\![1, i]\!], u.j \in Pos(t).$$

Prefixial sets are also termed *tree domains*. A tree such that $\mathcal{P}os(t) = \{\epsilon\}$ is called an *empty tree*.

A labeled tree t is thus a mapping from a tree domain $Pos(t)$ to U, where each node position is mapped to the label of the corresponding node. The label of node at position p in the tree t will be denoted $t@p$. This mapping is computed recursively as:

$$\begin{cases} f(t_1, \ldots, t_n)@\epsilon = f, \\ f(t_1, \ldots, t_n)@i.p = t_i@p \text{ for } i \in [\![1, n]\!]. \end{cases}$$

The set of variable positions in t is denoted by $\mathcal{VP}os(t)$ and $\mathcal{V}(t)$ is the set of variables in t. In the following, we will only consider trees in which these variables are located on leaves, i.e.:

$$\forall u \in U, [u \in \mathcal{P}os(t) \text{ and } \exists j, u.j \in \mathcal{P}os(t)] \Rightarrow t@u \notin V.$$

The *yield* of a tree t is the word in $L^\star$ defined as:

$$\begin{cases} \mathcal{Y}(l) = l, \\ \mathcal{Y}(f(t_1, \ldots, t_n)) = \mathcal{Y}(t_1).\ \ldots\ .\mathcal{Y}(t_n). \end{cases}$$

Example Let us consider the set of labels $L = \{a, b\}$ and the set of variables $V = \{x\}$. The tree $t_1 \in \mathcal{T}(U)$, defined as:

$$t_1@\epsilon = t_1@1 = t_1@2.2 = a,\ t_1@2 = t_1@2.1 = b \text{ and } t_1@2.3 = x,$$

and the tree $t_2 \in \mathcal{T}(U)$, defined as

$$t_2@\epsilon = t_2@1.1 = t@1.2 = a,\ t@1 = t@2 = b,$$

are displayed in Figure 15, where positions appear between parentheses. We have $\mathcal{VP}os(t_1) = \{x\}$, $\mathcal{VP}os(t_2) = \emptyset$, $\mathcal{Y}(t_1) = abax$, and $\mathcal{Y}(t_2) = aab$.

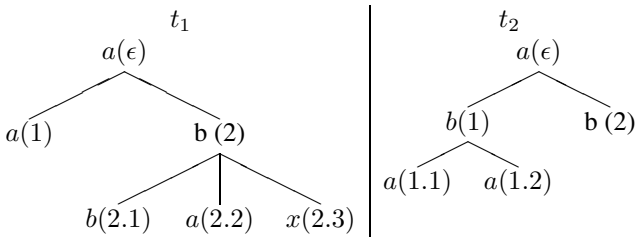


Figure 15: Two examples of trees

We are now in a position to introduce the notion of substitution.

Definition 10 (Substitution).
A substitution *is a pair $(v \leftarrow t')$, where $v \in V$ is a variable and $t' \in \mathcal{T}(U)$ is a tree. The* application *of the substitution $(v \leftarrow t')$ to a tree t is denoted $t \lhd_v t'$ and is computed inductively as:*

$$\begin{cases} v \lhd_v t' & = & t' \text{ if } v \in V, \\ l \lhd_v t' & = & l \text{ if } l \in L, \\ f(t_1, \ldots, t_n) \lhd_v t' & = & f(t_1 \lhd_v t', \ldots, t_n \lhd_v t'). \end{cases}$$

In other words, the application of a substitution consists in replacing *each leaf* of t labeled by v by the tree t'.

Example Continuing with the example trees of Figure 15, the application of substitution $(x \leftarrow t_2)$ to t_1 yields $t_1 \lhd_x t_2 = t_3$ (see Figure 16).

A factorization of a tree t is a sequence of substitutions whose final result is t. Put in formal terms:

Definition 11 (Factorization).
A factorization *of a tree $t \in \mathcal{T}(U)$ is a pair $[s, v]$, where s is a sequence of trees $(t_1, \ldots, t_n)$ and where v is a sequence of variables $(v_1, \ldots, v_{n-1})$ such that:*

- $t_1 \lhd_{v_1} \ldots \lhd_{v_{n-1}} t_n = t$;
- $\forall i \in [\![1, n]\!], |\{p \in Pos(t'_i) |\ t'_i@p = v_i\}| = 1,$ *where* $t'_i = t_1 \lhd_{v_1} \ldots \lhd_{v_{i-1}} t_i$.

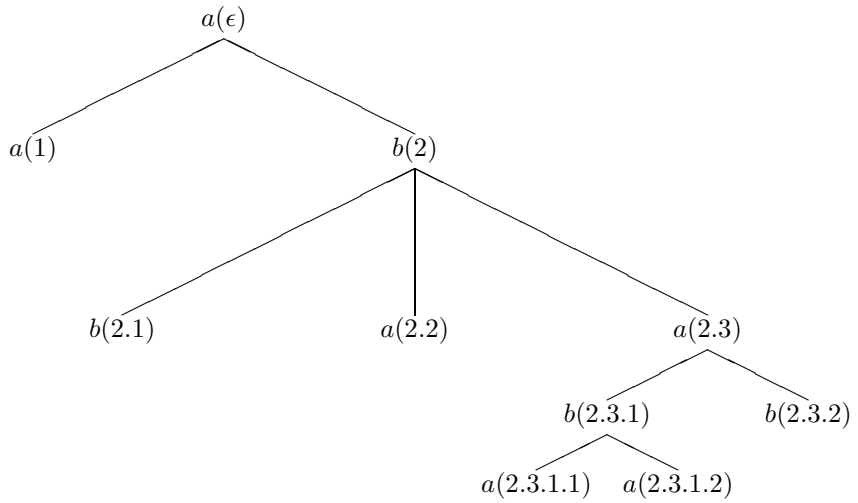


Figure 16: The result t_3 of the substitution $t_1 \lhd_x t_2$

The second condition expresses an essential restriction in our model: each substitution in the factorization must operate exactly on one node. In the following, we will consider two different notations for factorizations.

Two factorizations that only differ in the naming of variables can be considered equivalent, which is formalized as follows. Let $m : V_1 \to V_2$ be a bijective renaming of variables; by extension, $m(t)$ denotes the tree t in which each variable v of V_1 has been replaced by the corresponding variable $m(v)$ in V_2. For a factorization $[s, v]$ with $s = (t_1, \ldots, t_n)$ and $v = (v_1, \ldots, v_{n-1})$, we will also note

$$m([s, v]) = [(m(t_1), \ldots, m(t_n)), (m(v_1), m(v_2), \ldots, m(v_{n-1}))].$$

Since the renaming of variables is bijective, if $[s, v]$ is a factorization of t, then $m([s, v])$ is also a factorization of t. Moreover, if we write $[s, v] \sim [s', v']$ if and only if there exists a bijective renaming of variables m from $\{v_i\}_{i\in[\![1,n-1]\!]}$ to $\{v'_i\}_{i\in[\![1,n-1]\!]}$ such that $m([s, v]) = [s', v']$, then $\sim$ defines an equivalence relation.

In the following, we consider two different *projection maps* to refer to an equivalence class of factorizations using a particular representative of this class. In the first one, termed the "N notation", variables are denoted by their positions in the factorization, i.e. for a factorization $[s, v]$ with $s = (t_1, \ldots, t_n)$ and $v = (v_1, \ldots, v_{n-1})$, we have $\forall i \in [\![1, n-1]\!], N(v_i) = i$, i.e.:

$$N([s, v]) = [(N(t_1), \ldots, N(t_n)), (1, 2, \ldots, n-1)].$$

In the second one, termed the "P notation", variables are denoted by the (unique) position in the tree where the substitution takes place, i.e.:

$$P([s, v]) = [(P(t_1), \ldots, P(t_n)), (p_1, p_2, \ldots, p_{n-1})],$$
$$\text{with } \forall i \in [\![1, n-1]\!], (t_1 \lhd_{v_1} \ldots \lhd_{v_{i-1}} t_i)@p_i = v_i.$$

We will say that a factorization $[s, v]$ is an N-factorization (resp. a P-factorization) if $N([s, v]) = [s, v]$ (resp. $P([s, v]) = [s, v]$). By construction, there is only one N-factorization and one P-factorization per equivalence class.

5.2 Analogical proportions between trees

Provided with the basic notions of substitution and factorization, we first introduce, and then discuss, a general definition of proportions between labeled trees.

Definition 12 (Analogical proportion between trees).
For $(x, y, z, t) \in \mathcal{T}(U)^4$, we have $x : y :: z : t$ if and only if there exist four factorizations $[s_x, v]$, $[s_y, v]$, $[s_z, v]$, $[s_t, v]$ of x, y, z, and t respectively, such that:

$$\begin{aligned} s_x &= (x_1, \ldots, x_n), \\ s_y &= (y_1, \ldots, y_n), \\ s_z &= (z_1, \ldots, z_n), \\ s_t &= (t_1, \ldots, t_n), \end{aligned}$$

with $\forall i \in [\![1, n]\!], (y_i, z_i) \in \{(x_i, t_i), (t_i, x_i)\}$. As an additional restriction, we impose that the variables involved in the proportion appear in the same factors, i.e.

$$\forall i \in [\![1, n]\!], \mathcal{V}(x_i) = \mathcal{V}(y_i) = \mathcal{V}(z_i) = \mathcal{V}(t_i).$$

This definition is a direct adaptation of definition 2, in which subtrees are combined via substitutions. However, instead of using only one operator, we consider here a set of operators (one for each variable) and require that the variables involved in the four factorizations be identical. The first merit of this definition is thus its consistency with our general understanding of proportions.

It can moreover be shown that, given this definition, the "syntactic" proportion illustrated in section 2.2 holds (cf. Figure 1), as well as the more morphologically-oriented example in Figure 17 (a). Figure 17 (b) uncovers the factorizations underlying this proportion. See also Figure 18 for a detailed view of the example in Figure 1.

5.3 Computing proportions between trees

This section is devoted to a study of the computational implications of this definition. We first present an exact algorithm for deciding whether four trees form a proportion and explain how to turn it into a solver. This algorithm however has an (at least) exponential worst case complexity. We thus consider approximate algorithms for verifying and solving proportional equations, which essentially put to use various tree linearization procedures and the available verifier and solver for proportions between sequences. These approximations are only valid for a more restrictive notion of proportions between trees.

5.3.1 Validating proportions between trees

The algorithm we propose for validating proportions between trees is a *constructive* algorithm, which tries to build factorizations supporting the proportion. This algorithm is a two-step process:

1. For each tree α in $\{x, y, z, t\}$, create a finite-state automaton A_α representing all the P-factorizations of α (cf. Algorithm 1).

2. Search for compatible factorizations within these automata, where compatibility is defined as in definition 12. This step puts to use some of the algorithms introduced in Section 3.4.3 to compute and validate proportions between words.

These steps are detailed in the following paragraphs.

Algorithm 1: `ConstructAutomaton`(t,p)

Input : A tree $t = l(t_1, \ldots, t_n)$
Output: An automata $A = \langle Q, E, i, f \rangle$

1 $i \leftarrow \{\epsilon\}$
2 $f \leftarrow \emptyset$
3 **if** $n = 0$ **then**
4 $\quad$ **return** $\langle \{i, f\}, \{(i, i, \epsilon \rightarrow \epsilon), (i, f, \epsilon \rightarrow l)\}, i, f \rangle$
5 $Q \leftarrow \{i\}$
6 $E \leftarrow \{(i, i, \epsilon \rightarrow \epsilon)\}$
7 **for** $j \leftarrow 1$ **to** n **do**
8 $\quad A_j = \langle Q_j, E_j, i_j, f_j \rangle \leftarrow$ `ConstructAutomaton(`t_j`)`
9 **for** $(q_1, \ldots, q_n) \in Q_1 \times \cdots \times Q_n$ **do**
10 $\quad q \leftarrow 1.q_1 \cup \ldots \cdots \cup n.q_n$
11 $\quad Q \leftarrow Q \cup \{q\}$
12 $\quad$ **for** $j \leftarrow 1$ **to** n **do**
13 $\quad\quad$ **for** $(q_j, q'_j, p \rightarrow u) \in E_j$ **do**
14 $\quad\quad\quad q' \leftarrow 1.q_1 \cup \cdots \cup (j-1).q_{j-1} \cup j.q'_j \cup (j+1).q_{j+1} \cup \cdots \cup n.q_n$
15 $\quad\quad\quad E \leftarrow E \cup (q, q', j.p \rightarrow \mathrm{add\_pref}(j, u))$
16 $\quad E \leftarrow E \cup (i, q, \epsilon \rightarrow \gamma^{-1}(q))$
17 **return** $\langle Q, E, i, f \rangle$

Construction of the automata Algorithm 1 is essentially a recursive exploration of all the possible decompositions of tree α in subtrees. Each state defines a set of positions where the variables are located, and where substitution will take place. Each transition corresponds to a substitution, which replaces a variable with a tree. If t is a leaf node, there are only two states, an initial state i associated to the variable position ϵ, and a final state f associated with the empty set $\emptyset$ – which means there is no variable to substitute. A (loop) transition on i substitutes ϵ with itself; a transition between i and f rewrites ϵ into l, the label of the unique node of t (Lines 1-4). If the tree t is not empty, we initialize the set of states with the initial state (associated to the variable position ϵ), and the set of transitions with the loop over this initial state (Lines 5-6). We then build the automata A_j for all the children t_j of the root of t (Lines 7-8); indeed, a substitution can occur in any of these children, leaving the rest untouched, and we thus have to consider all the possible combinations of states (Lines 9-11). Transitions are built upon the set of substitutions (Lines 13-15) that occur in each child (loop in Line 12); $\mathrm{add\_pref}(j, u)$ denotes the tree u in which the prefix j is added to all the variables (so as to turn a subtree of t_j into a proper subtree of t). Finally, a substitution can directly rewrite ϵ, hence Line 16.[16] An example of a tree t and the associated

[16] Each state q can be associated to a tree, which is the result of the applications of the substitutions leading to q; $\gamma^{-1}(q)$ denotes this tree. See Appendix B for a formal and detailed explanation.

A_t is displayed in Figure 19. The proof of the following proposition can be found in Appendix B.

Proposition 12.
The successful paths in A_t are the P-factorizations of t.

Uncovering consistent factorizations In order to validate a proportion, we now have to find factorizations that meet the requirements imposed by definition 12, i.e. factorizations involving alternating substitutions. In order to do this, our algorithm proceeds in two steps:[17]

- The first step extracts quadruples of P-factorizations s_x, s_y, s_z, s_t, with $\forall \alpha \in \{x, y, z, t\}$, $s_\alpha = [(\alpha_1, \ldots, \alpha_n), (p_{\alpha_1}, \ldots, p_{\alpha_{n-1}})]$, such that:

  $$\forall i \in [\![1, n]\!], d(x_i) : d(y_i) :: d(z_i) : d(t_i),$$

  where d denotes the operation which replaces all variables by a dummy symbol DUM. This extraction is achieved using the validation algorithm for strings introduced in Section 3.4.3.[18] Indeed, if this condition is verified, the paths c_x, c_y, c_z, c_t defined for $\alpha \in \{x, y, z, t\}$ as $c_\alpha = (\mathrm{DUM} \rightarrow \alpha_1, \ldots, \mathrm{DUM} \rightarrow \alpha_n)$ form a valid proportion between strings (of "generalized" substitutions), and conversely.

- The second step consists in checking that it is possible to find a consistent assignment of the variables, consistent meaning here that the substitutions involved in the factorizations actually rewrite the same variables at the same time. Given the P-factorizations s_x, s_y, s_z, s_t, and noting for $\alpha \in \{x, y, z, t\}$, $N(s_\alpha) = [(\alpha'_1, \ldots, \alpha'_n), (1, \ldots, n-1)]$, this is simply achieved by checking out that $\forall i \in [\![1, n]\!]$, $(y'_i, z'_i) \in \{(x'_i, t'_i), (t'_i, x'_i)\}$, and $\mathcal{V}(x'_i) = \mathcal{V}(y'_i) = \mathcal{V}(z'_i) = \mathcal{V}(t'_i)$.

The correctness and completeness of the algorithm are proved as follows.

Proof. **Correctness.** By construction, the validation algorithm provides us with four factorizations f_x, f_y, f_z, and f_t respectively of x, y, z, and t such that

$$\forall \alpha \in \{x, y, z, t\}, N(f_\alpha) = [(\alpha'_1, \ldots, \alpha'_n), (1, \ldots, n-1)],$$

with $\forall i \in [\![1, n]\!]$, $(y'_i, z'_i) \in \{(x'_i, t'_i), (t'_i, x'_i)\}$, and $\mathcal{V}(x'_i) = \mathcal{V}(y'_i) = \mathcal{V}(z'_i) = \mathcal{V}(t'_i)$. We thus have $x : y :: z : t$.
Completeness. By definition of proportion between trees, there exist four factorizations $[s_x, v]$, $[s_y, v]$, $[s_z, v]$, $[s_t, v]$ of x, y, z, and t respectively, such that

$$\forall \alpha \in \{x, y, z, t\}, s_\alpha = (\alpha_1, \ldots, \alpha_n),$$

[17] A proportion between trees involves substitutions that rewrite the same variables at the same time. Consequently, when looking for proportions, N-factorizations, in which variables are represented by their positions in the *factorization*, are more adapted than P-factorizations, in which variables are represented by their positions in the *trees*. However, it is more difficult to express the set of N-factorizations in an efficient way than it is for P-factorizations, hence the introduction of the two steps in our approach. First, candidate factorizations are extracted from a compact representation of the set of P-factorizations, then a consistency check on variables is performed.

[18] In particular, this step requires to build the shuffle products $A'_x \bullet A'_t$ and $A'_y \bullet A'_z$ and the intersection $A'_x \bullet A'_t \cap A'_y \bullet A'_z$, where A'_α denotes the automaton A_α in which the variables have been "generalized" using a dummy symbol.

with $\forall i \in [\![1,n]\!], (y_i, z_i) \in \{(x_i, t_i), (t_i, x_i)\}$ and $\mathcal{V}(x_i) = \mathcal{V}(y_i) = \mathcal{V}(z_i) = \mathcal{V}(t_i)$. By construction, for $\alpha \in \{x, y, z, t\}$, the automaton A_α contains the P-factorization $P([s_\alpha, v])$, which we note $P([s_\alpha, v]) = [(\alpha'_1, \ldots, \alpha'_n), (p_{\alpha'_1}, \ldots, p_{\alpha'_{n-1}})]$. We have:

$$\begin{aligned}
&\forall i \in [\![1,n]\!], (y_i, z_i) \in \{(x_i, t_i), (t_i, x_i)\},\\
\text{so } &\forall i \in [\![1,n]\!], (P(y_i), P(z_i)) \in \{(P(x_i), P(t_i)), (P(t_i), P(x_i))\},\\
\text{and } &\forall i \in [\![1,n]\!], (d(P(y_i)), d(P(z_i))) \in \{(d(P(x_i)), d(P(t_i))), (d(P(t_i)), d(P(x_i)))\}.
\end{aligned}$$

The validation algorithm will thus find the paths associated to $P([s_\alpha, v])$ for $\alpha \in \{x, y, z, t\}$ and the first step of the algorithm will succeed. The last thing to check is:

$$\begin{aligned}
\forall i \in [\![1,n]\!], &(N(y_i), N(z_i)) \in \{(N(x_i), N(t_i)), (N(t_i), N(x_i))\}\\
&\text{and } \mathcal{V}(N(x_i)) = \mathcal{V}(N(y_i)) = \mathcal{V}(N(z_i)) = \mathcal{V}(N(t_i)),
\end{aligned}$$

which, again, is a direct consequence of $\forall i \in [\![1,n]\!], (y_i, z_i) \in \{(x_i, t_i), (t_i, x_i)\}$ and $\mathcal{V}(x_i) = \mathcal{V}(y_i) = \mathcal{V}(z_i) = \mathcal{V}(t_i)$. □

Complexity There is one state in A_α for each antichain of $\mathcal{P}os(\alpha)$. In the general case, the number of states is thus (at least) exponential with respect to the number of internal nodes; for a tree $\alpha = l(\alpha_1, \ldots, \alpha_n)$, the number of nodes $\pi(\alpha)$ is computed using the following recurrence, which can be established directly from Algorithm 1:

$$\left\{ \begin{array}{lcl} \pi(l) & = & 2 \\ \pi(l(\alpha_1, \ldots, \alpha_n)) & = & \prod_{1 \leq i \leq n}(1 + \pi(\alpha_i)). \end{array} \right.$$

In the case of a tree made of one root directly dominating $n-1$ leaves, this number is $\pi(\alpha) = 3^{n-1}$. In the case of a balanced tree α of width k (the width is the constant branching factor of the internal nodes in α), we have $\pi(\alpha) \leq k^{C \times n}$, where C is a constant which depends only on k.

The complexity of the extraction and verification step in the worst case is thus at least exponential in the number of nodes of the trees. It is up to future work to establish whether this decision problem is NP-complete.

5.3.2 Solving analogical equations between trees

Based on the algorithm presented above, it is possible to derive a solver for analogical equations between trees. Given three trees x, y, z, this solver constructs the automata A_x, A_y, A_z, then build the shuffle product $A'_y \bullet A'_z$, and use the complementary operation with respect to A'_x to obtain $A'_y \bullet A'_z \backslash A'_x$, where A'_α denotes the automaton A_α in which the variables have been "generalized" using a dummy symbol. This gives a set of candidate sequences of substitutions. For each candidate, as it is the case for verification, we need to perform a check on variables to finally get the set of correct solutions for the equation $x : y :: z :?$. The complexity of this solver is established in the same manner as for the verification procedure.

5.3.3 Approximative verification and solving algorithms

In this section, we present approximate algorithms aimed at validating analogical proportions and solving analogical equations, which rely on tree linearization procedures.

These approximations are based on a more restrictive notion of proportions between trees, which is detailed below.

A tree linearization is an injective application from a set of trees to some free monoid; it establishes a unique correspondence between a tree and a string. We consider here two different tree linearization procedures. The first is based on parenthesized expressions; the second one is based on a prefix traversal of the tree, along which each node label and arity are collected. These linearizations are more formally defined as follows.

Definition 13 (Parenthesized expressions).
Let U be a set of labels and "(", ")" two symbols not in U. $\bar{U}$ denotes $U \cup \{(,)\}$. The parenthesized expression *of a tree t is the word $p(t) \in \bar{U}^\star$ inductively defined as:*[19]

$$p(f(t_1, \ldots, t_n)) = (.f.p(t_1). \ \ldots \ .p(t_n).).$$

Example By considering the tree t_3 of Figure 16, we have:

$$p(t_3) = (a(a)(b(b)(a)(a(b(a)(a))b))).$$

Definition 14 (Arity-based linearization).
The arity-based linearization *of a tree t is the word $ab(t) \in (L \times \mathbb{N})^\star$ inductively defined as:*

$$ab(f(t_1, \ldots, t_n)) = (f, n).ab(t_1). \ \ldots \ .ab(t_n).$$

Example By again considering t_3, we have:

$$ab(t_3) = (a,2)(a,0)(b,3)(b,0)(a,0)(a,2)(b,2)(a,0)(a,0)(b,0).$$

In the following, we consider proportions in which factorizations involve variables appearing in the same relative lexicographic order in the four trees, i.e. for a proportion $x : y :: z : t$ supported by the factorizations $[(\alpha_1, \ldots, \alpha_n), (v_1, \ldots, v_{n-1})]$ for $\alpha \in \{x, y, z, t\}$, we add the following constraints:

$$\begin{gathered}
\forall \alpha, \alpha' \in \{x, y, z, t\}, \forall i \in [\![1, n]\!], \forall v_j, v_k \in \{v_1, \ldots, v_n\}, \\
\text{if } (\alpha_1 \lhd_{v_1} \ldots \lhd_{v_{i-1}} \alpha_i)@n_j = v_j, \ (\alpha_1 \lhd_{v_1} \ldots \lhd_{v_{i-1}} \alpha_i)@n_k = v_k, \\
\text{and } (\alpha'_1 \lhd_{v_1} \ldots \lhd_{v_{i-1}} \alpha'_i)@n'_j = v_j, \ (\alpha'_1 \lhd_{v_1} \ldots \lhd_{v_{i-1}} \alpha'_i)@n'_k = v_k, \\
\text{then } n_j < n_k \Leftrightarrow n'_j < n'_k.
\end{gathered}$$

A proportion (between trees) satisfying these conditions will be noted $x : y \overset{C}{::} z : t$ (C standing here for "constrained"). The rationale behind this additional constraint is that, in the case it is verified, we can relate proportions between trees and proportions between their linearizations. In particular, if a proportion between trees holds, then this implies that there is also a proportion between their linearizations (seen as words). By doing so, we lose some of the expressiveness of proportions between trees (for example, a passive/active opposition cannot be captured); however, it makes the application of efficient algorithms possible, as demonstrated by the following result.

Proposition 13.
For $(x, y, z, t) \in (\mathcal{T}(U))^4$, we have:

$$x : y \overset{C}{::} z : t \Rightarrow p(x) : p(y) :: p(z) : p(t).$$

[19] The concatenation operation is explicitly noted for readability.

Proof. First, let us remark that for a tree t, and a variable v appearing exactly once in t, if $p(t) = u_1 v u_2$, then for any tree t', $p(t \lhd_v t') = u_1 p(t') u_2$.

The proof is established by induction on the degree n of the proportion. We will also prove that for any proportion $x : y \overset{C}{::} z : t$ of degree n, if $\{v_1, \ldots, v_k\}$ is the set of variables appearing in x,y,z, and t, in lexicographic order, we have:

$$\forall \alpha \in \{x, y, z, t\}, \exists (u_{\alpha_i})_{i \in [\![1, k+1]\!]} \in \bar{U}^\star, \text{ so that } p(\alpha) = u_{\alpha_1}\ v_1\ u_{\alpha_2} \ldots\ v_k\ u_{\alpha_{k+1}},$$
$$\text{with } \forall i \in [\![1, k+1]\!],\ u_{x_i} : u_{y_i} :: u_{z_i} : u_{t_i}.$$

Let $(x, y, z, t) \in (\mathcal{T}(U))^4$ such that $x : y \overset{C}{::} z : t$. By definition, there exist four factorizations $[s_x, v]$, $[s_y, v]$, $[s_z, v]$, $[s_t, v]$ of x, y, z, and t respectively, such that: $s_\alpha = (\alpha_1, \ldots, \alpha_n)$ for $\alpha \in \{x, y, z, t\}$, with $\forall i \in [\![1, n]\!], (y_i, z_i) \in \{(x_i, t_i), (t_i, x_i)\}$ and $\mathcal{V}(x_i) = \mathcal{V}(y_i) = \mathcal{V}(z_i) = \mathcal{V}(t_i)$, and the additional specific constraint introduced above for $\overset{C}{::}$. If $n = 1$, the proportions are atomic, and since the variables appear in the four trees in the same lexicographic order, it is also the case for the variables in their linearizations. The implication is thus verified for $n = 1$.

We now assume that it also holds for any proportion of degree $k < n$. For α in $\{x, y, z, t\}$, we set $\alpha' = \alpha_1 \lhd_{v_1} \ldots \lhd_{v_{n-2}} \alpha_{n-1}$ and $v = v_{n-1}$, so we can write $\alpha = \alpha' \lhd_v \alpha_n$. The proportion $x' : y' :: z' : t'$ clearly holds and its degree is at most $n - 1$. The induction hypothesis thus yields:

$$\forall \alpha \in \{x, y, z, t\}, \exists (u_{\alpha_i})_{i \in [\![1, k+1]\!]} \in \bar{U}^\star, \text{ so that } p(\alpha') = u_{\alpha_1}\ \ldots\ v\ \ldots\ u_{\alpha_{k+1}},$$
$$\text{with } \forall i \in [\![1, k+1]\!],\ u_{x_i} : u_{y_i} :: u_{z_i} : u_{t_i}.$$

We then have $p(\alpha) = u_{\alpha_1}\ \ldots\ p(\alpha_n)\ \ldots\ u_{\alpha_{k+1}}$. Moreover, since the variables in x_n, y_n, z_n, and t_n appear exactly once and in the same relative order, we have $p(x) : p(y) :: p(z) : p(t)$, as well as the property added to the induction hypothesis. □

Proposition 14.
For $(x, y, z, t) \in (\mathcal{T}(U))^4$, we have:

$$x : y \overset{C}{::} z : t \Rightarrow ab(x) : ab(y) :: ab(z) : ab(t).$$

Proof. The demonstration is directly obtained by replacing $p(w)$ by $ab(w)$ in the proof of proposition 13. □

More generally, the proposition is true whenever the linearization respects the lexicographic order of the nodes.

These results warrant the use of an approximate verifier or solver based on tree linearization procedures. Moreover, it is possible to stack several verifiers or solvers based on various linearizations in order to get closer to the exact solutions. We can also use these procedures as a generating device, which proposes analogical proportions which will then be validated or filtered by the exact algorithm presented in the previous section.

5.4 Summary

In this section, we have defined two alternative notions of proportions between labeled trees. The first one is more general and covers a wider range of linguistic phenomena. Notably, it relates structures in which constituents are moved from one position in the

tree to another, as is the case in active/passive transformations. Based on this first definition, we have presented an exponential algorithm for validating proportions, and for solving equations.

The second definition is more restrictive, as it requires that constituents should occur in the same positions in the trees. However, it lends itself to faster, albeit approximate, algorithms, which can be used in a two-step validations procedure.

Both definitions can be further generalized in a manner similar to the generalization discussed in Section 4: by replacing the condition that the labels occurring on leaf nodes should match exactly by the less stringent requirement that they define a proportion on sequences.

6 Conclusion and perspectives

6.1 Conclusion

In this paper, we have introduced a formal algebraic framework for the notion of analogical proportions. This framework applies to a large range of algebraic structures: semigroups, free monoids, lattices, languages, and sets of trees. This framework is based on the decomposition of objects into smaller parts that alternate. Depending on the actual structure, the generic definition we propose may be simplified, and the verification and solving procedures may be more or less expensive. We discussed the algorithmic complexity of these procedures in several cases. Efficient algorithms are available for strings, feature-structures, and sets. In the case of trees, we can build approximate algorithms upon various linearizations. This yields a computational model suitable for most structured representations commonly used in NLP applications. Moreover, the examples of proportions we planned to model are covered by our definitions.

This framework has been implemented within the software ALANIS[20] (A Learning-by-ANalogy Inferencer for Structured data): learning devices that are based on analogical proportions can benefit from this implementation and are now applicable to a wide range of linguistics representations and NLP tasks. The results of several experiments using this software are actually reported by Stroppa and Yvon [2005], Stroppa [2005], Stroppa and Yvon [2006], for the following tasks: (i) grapheme-to-phoneme conversion, (ii) inflectional analysis of word forms, and (iii) derivational analysis of word forms. In the first task, inputs and outputs are strings (sequences of graphemes and phonemes). In the second one, we look for the lemma and the morphological features associated to a given word form; inputs are strings and outputs are pairs of string and (flat) feature structures. The goal of the third task is to find the hierarchical decomposition of a wordform into morphemes; inputs are strings and outputs are labeled trees. These experimental results show that state-of-the-art performances can be achieved for these three tasks.

6.2 Perspectives

The work reported in this paper has deliberately concentrated on the computation of proportions: this is, of course, only half of the story and a lot of efforts are required to put these algorithms to use in realistic machine learning applications. This aspect of

[20] This software is freely available from `http://www.computing.dcu.ie/~nstroppa/index.php?page=softwares`.

the work is still on-going, even if promising results have been obtained in a variety of applications and for various languages (see the references cited above).

Current development is active on two separate fronts. On the experimental front, we intend to widen the range of applications of analogical learning, by considering more language types (especially languages exhibiting complex and/or non concatenative morphological phenomena), and by considering novel natural language processing tasks, such as, for instance, word sense disambiguation, which seems to be a promising test-bed for analogical approaches (see the discussion in [Pirrelli and Yvon, 1999] and the references cited therein).

On the more theoretical front, the main computational burden of a machine learning approach based on analogical proportions remains the search, in the database of known instances, of candidate analog triples on which to ground the inference procedure. A brute force approach would require to evaluate a number of proportions that grows as the cube of the number of training instances, which, given the complexity of the algorithms discussed in this paper, and the typical size of dictionaries or tree-banks, is prohibitive. We believe that the search step could be made considerably faster if we could somehow restrict this search to a subset of *promising* triples. In other terms, what we need are procedures allowing to *rank* candidates based on some numerical scores. Propositions for defining a gradual notion of analogical proportions have been made in [Yvon et al., 2004] and our next step will be to reevaluate these propositions and see how they can help improve the search procedure.

6.3 An algorithmic vision of analogical proportions

As a concluding note, we would finally like to emphasize the relationship between our work and some conceptions of analogical proportions that are prevalent in some Artificial Intelligence (AI) circles. In particular, we would like to show that our framework can be extended to take into account more complex proportions, such as those studied in AI.

In our framework, proportions are mainly based on the notions of decomposition and alternation (cf. Section 3). This is suited to the "simple" case of semigroup and its derivatives, where only one operator is available. However, two directions have already been explored, which enhance the expressiveness of the approach: (i) the use of multi-level proportions, which stack structures recursively, and (ii) the addition of operators, which enrich the composition mechanism; these kinds of extensions have been introduced to handle languages (cf. Section 4.2) and have also proved necessary to handle the case of trees (cf. Section 5).

By using unrestricted operators and unlimited recursive schemes, we are actually heading towards a situation where proportions can be defined for any computable objects. The decomposition of an object is simply an algorithmic procedure computing this object: $u(x_1, \ldots, x_n) = x$ is an "algorithmic factorization" u_x of x, where u is an algorithm and $x_1, \ldots, x_n$ some parameters. By convention, we will note $u_x(0) = u$ and $\forall i \in [\![1, n]\!]$, $u_x(i) = x_i$. A definition of analogical proportion which applies to any computable object can be derived from these considerations.

Definition 15 (Analogical proportions between computable objects).
For a quadruple (x, y, z, t), we have $x : y :: z : t$ if and only if there exist four algorithmic factorizations u_x, u_y, u_z, u_t of x, y, z, and t respectively, such that:

$$\forall i \in [\![0, n]\!], (u_y(i), u_z(i)) \in \{(u_x(i), u_t(i)), (u_t(i), u_x(i))\}$$

This definition is completely general and consistent with the work of Schmid et al. [2003]. It can also be used to model complex analogies: most of the examples studied by Hofstadter and the Fluid Analogies Research Group [1995] can be expressed in this framework. For example, in order to model $abc : abd :: ijk : ijl$, it is sufficient to write $u(a_1, a_2, a_3, a_4) = a_1.a_2.(a_3 + a_4)$, which yields:

$$u(a, b, c, 0) = abc, \;\; u(a, b, c, 1) = abd, \;\; u(i, j, k, 0) = ijk, \;\; u(i, j, k, 1) = ijl.$$

The relationship between the letters a, b, and c can also be exploited: if we note $u(a_1, a_2) = a_1.(a_1 + 1).(a_1 + 2 + a_2)$, we have

$$u(a, 0) = abc, \; u(a, 1) = abd, \; u(i, 0) = ijk, \; u(i, 1) = ijl.$$

This also applies to proportions such as $abc : abd :: mrrjjj : mrrkkk$.

The algorithms used in the case of strings (cf. Section 3.4), are very simple: they can only concatenate strings. The less strings they are concatenating, the less complex they are considered (cf. notion of degree), and better is the proportion. In the algorithmic context, the (Kolmogorov) algorithmic complexity (as described e.g. by Li and Vitányi [1997]) seems to be a reasonable criterion to qualify a proportion:[21] the less complex the algorithms involved in proportion, the better the proportion, and the more likely this proportion will match the intuition. In other words, we have recasted the problem of recognizing and verifying analogies into the problem of finding algorithms with small complexities. We do not plan to develop our work along those lines any further and intend to continue focusing on those specific structures that are commonly used in NLP applications; it is however comforting to realize that our work is consistent with general purpose, cognitively-oriented visions of analogical proportions.

[21] Describing objects thanks to the algorithms that can generate them is the basis of the algorithmic complexity.

References

Stephen R. Anderson. *A-Morphous Morphology*. Cambridge University Press, 1992.

Aristote. *Poétique*. Les Belles Lettres, Paris. (traduction de J. Hardy, 1990).

James P. Blevins. Word-based morphology. *Journal of Linguistics*, 42:531–573, 2006.

Leonard Bloomfield. *Le Langage*. Payot, 1970. 1st ed. New York, 1933.

Rens Bod. A computational model of language performance: Data oriented parsing. In *Proceedings of 14th International Conference on Computational Linguistics (COLING 1992)*, pages 855–859, Nantes, France, 1992.

Joan Bresnan. *Lexical-Functional Syntax*. Blackwell Publishers, Oxford, 2001.

Sabine Buchholz and Erwin Marsi. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL 2006)*, pages 149–164, New York, NY, 2006.

Bob Carpenter. *The Logic of Typed Feature Structures*. Number 32 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1992.

Noam Chomsky. *The logical structure of linguistic theory*. University of Chicago Press, Chicago, IL, 1975.

H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: `http://www.grappa.univ-lille3.fr/tata`, 1997. Release October, 1st 2002.

Ferdinand de Saussure. *Cours de linguistique générale*. Payot, Lausanne et Paris, 2e ed. 1995 edition, 1916.

T.G. Evans. A program for the solution of a class of geometric analogy intelligence test questions. In M. Minsky, editor, *Semantic Information Processing*, pages 271–353. MIT Press, Cambridge, MA, 1968.

Robert M. French. The computational modeling of analogy-making. *Trends in Cognitive Science*, 6(5):200–205, 2002.

Dedre Gentner, Keith J. Holyoak, and Boicho Kokinov, editors. *The Analogical Mind: Perspectives from Cognitive Science*. MIT Press, Cambridge, MA, 2001.

G. Grätzer. *Lattice Theory: First Concepts and Distributive Lattices*. W. H. Freeman, San Francisco, CA, 1971.

Robert R. Hoffman. Monster analogies. *AI Magazine*, pages 11–35, 1995.

Douglas R. Hofstadter and the Fluid Analogies Research Group, editors. *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought*. Basic Books, New York, NY, 1995.

Bipin Indurkhya. *Metaphor and cognition: an interactionist approach*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1992.

Esa Itkonen and Jussi Haukioja. A rehabilitation of analogy in syntax (and elsewhere). In András Kertész, editor, *Metalinguistik im Wandel: die kognitive Wende in Wissenschaftstheorie und Linguistik*, pages 131–177. Peter Lang, Frankfurt, Germany, 1997.

Nilendu G. Jani and Daniel S. Levine. A neural network theory of proportional analogy-making. *Neural Networks*, 13(2):149–183, 2000.

René Joseph Lavie. *Le locuteur analogique ou la grammaire mise à sa place*. PhD thesis, Université de Paris 10, France, 2003.

Yves Lepage. Solving analogies on words: an algorithm. In *Proceedings of the 36thAnnual Meeting of the Association for Computational Linguistics and 17thInternational Conference on Computational Linguistics*, volume 1, pages 728–735, Montréal, Canada, 1998.

Yves Lepage. Analogy + tables = conjugation. In *Proceedings of the International Conference on Applications of Natural Language to Data Bases (NLDB 1999)*, pages 197–201, Klagenfurt, Germany, 1999a.

Yves Lepage. Open set experiments with direct analysis by analogy. In *Proceedings of the 5thNatural Language Processing Pacific Rim Symposium (NLPRS 1999)*, pages 363–368, Beijing, China, 1999b.

Yves Lepage. Analogy and formal languages. In *Proceedings of the 6thconference on Formal Grammar and the 7thMeeting on Mathematics of Language (FG-MOL 2001)*, pages 373–378, Helsinki, Finland, 2001.

Yves Lepage. De l'analogie rendant compte de la commutation en linguistique. Habilitation à diriger les recherches, Grenoble, France, 2003.

Ming Li and Paul Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer-Verlag, New York, 2ndedition, 1997.

Sylvain Lombardy, Raphaël Poss, Yann Régis-Gianas, and Jacques Sakarovitch. Introducing Vaucanson. In *Proceedings of the 8thInternational Conference on Implementation and Application of Automata (CIAA 2003)*, pages 96–107, Santa Barbara, CA, 2003.

Sylvain Lombardy, Yann Régis-Gianas, and Jacques Sakarovitch. Introducing Vaucanson. *Theoretical Computer Science*, 328:77–96, 2004.

Peter H. Matthews. *Morphology*. Cambridge University Press, Cambridge, 1974.

Peter H. Matthews. *Syntax*. Cambridge Textbooks in Linguistics. Cambridge University Press, 1981.

Igor A. Mel'čuk. *Dependency Syntax: Theory and Practice*. State University of New York Press, 1988.

Vito Pirrelli and François Yvon. The hidden dimension: paradigmatic approaches to data-driven natural language processing. *Journal of Experimental and Theoretical Artificial Intelligence, Special Issue on Memory-Based Language Processing*, 11: 391–408, 1999.

Carl Pollard and Ivan A. Sag. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago, 1994.

Yann Régis-Gianas and Raphaël Poss. On orthogonal specialization in C++: Dealing with efficiency and algebraic abstraction in Vaucanson. In *Proceedings of the Parallel/High-performance Object-Oriented Scientific Computing (POOSC 2003)*, pages 71–82, Darmstadt, Germany, 2003.

Jacques Sakarovitch. *Éléments de théorie des automates*. Vuibert, Paris, 2003.

Ute Schmid, Helmar Gust, Kai-Uwe Kühnberger, and Jochen Burghardt. An algebraic framework for solving proportional and predictive analogies. In F. Schmalhofer, R. Young, and G. Katz, editors, *Proceedings of the European Conference on Cognitive Science (EuroCogSci 2003)*, pages 295–300, Osnabrück, Germany, 2003. Lawrence Erlbaum.

Stuart M. Shieber. Synchronous grammars as tree transducers. In *Proceedings of the Seventh International Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+ 7)*, pages 88–95, Vancouver, Canada, 2004.

Nicolas Stroppa. *Définitions et caractérisations de modèles à base d'analogies pour l'apprentissage automatique des langues naturelles*. PhD thesis, École Nationale Supérieure des Télécommunications, Paris, 2005.

Nicolas Stroppa and François Yvon. An analogical learner for morphological analysis. In *Proceedings of the 9thConference on Computational Natural Language Learning (CoNLL 2005)*, pages 120–127, Ann Arbor, MI, 2005.

Nicolas Stroppa and François Yvon. Du quatrième de proportion comme principe inductif : une proposition et son application à l'apprentissage de la morphologie. *Traitement Automatique des Langues*, 47(2), 2006.

Gregory Stump. *Inflectional Morphology*. Cambridge University Press, Cambridge, MA, 2001.

William H. Wilson, Deborah J. Street, and Graeme S. Halford. Solving proportional analogy problems using tensor product networks with random representations. In *Proceedings of the IEEE International Conference on Neural Networks (ICNN '95)*, pages 2971–2975, Perth, Australia, 1995.

François Yvon. Paradigmatic cascades: a linguistically sound model of pronunciation by analogy. In *Proceedings of the 35thAnnual Meeting of the Association for Computational Linguistics and the 8thAnnual Meeting of the European Chapter of the Association for Computational Linguistics*, pages 248–435, Madrid, Spain, 1997.

François Yvon. Pronouncing unknown words using multi-dimensional analogies. In *Proceedings of the European Conference on Speech Application and Technology (Eurospeech)*, volume 1, pages 199–202, Budapest, Hungary, 1999.

François Yvon. Finite-state machines solving analogies on words. Technical Report D008, École Nationale Supérieure des Télécommunications, Paris, France, 2003.

François Yvon, Nicolas Stroppa, Arnaud Delhay, and Laurent Miclet. Solving analogies on words. Technical Report D005, École Nationale Supérieure des Télécommunications, Paris, France, 2004.

A Proof of proposition 10

Proof. Let $x : y :: z : t$ be an analogical proportion with

$$D = D(x : y :: z : t), O = O_\epsilon(x : y :: z : t,\ x) \text{ and } D > O > 1.$$

By hypothesis, there exist some factorizations (f_x, f_y, f_z, f_t) of size n and an interval $J = [\![i, j]\!]$ such that $f_x(J) = \epsilon$, $|\,J\,| = O$ and for all interval $I \subseteq [\![1, n]\!]$ with $|\,I\,| > O$, $f_x(I) \neq \epsilon$. Since $D > O$, one factor in x is not empty and we have either $i > 1$ or $j < D$. For symmetry reasons, we can assume without loss of generality that $i > 1$. The proof is achieved by building new factorizations of x, y and z obtained by permuting the factors at position i and $i + 1$ in x, y and z. For $\alpha \in \{x, y, z, t\}$, we define

$$\begin{array}{lll} \hat{f}_\alpha(k) & = f_\alpha(k) & \text{if } k < i - 1, \\ \hat{f}_\alpha(i-1) & = f_\alpha(i-1)f_\alpha(i+1), & \\ \hat{f}_\alpha(i) & = f_\alpha(i), & \\ \hat{f}_\alpha(k) & = f_\alpha(k+1) & \text{if } i < k < D. \end{array}$$

Since $|\,J\,| > 1$, we have $i+1 \in J$ and $f_x(i+1) = \epsilon$. Moreover, for symmetry reasons, we can assume that $(f_x(i), f_t(i)) = (f_y(i), f_z(i))$ and $(f_x(i+1), f_t(i+1)) = (f_z(i+1), f_y(i+1))$. In particular, since $f_x(i) = f_y(i) = \epsilon$ and $f_x(i+1) = f_z(i+1) = \epsilon$, we have $f_\alpha(i+1)f_\alpha(i) = f_\alpha(i-1)f_\alpha(i)$ for $\alpha \in \{x, y, z\}$, and permuting these chunks yields proper factorizations of x, y and z. The result of this construction is a factorization $\hat{f}_t$ for a word $\hat{t}$ such that $x : y :: z : \hat{t}$ with

$$D(x : y :: z : \hat{t}) = |\,\hat{f}_x\,| = |\,f_x\,| - 1 = D - 1.$$

Moreover, the "local" degree of these new factorizations has decreased by 1, so the equality $O_\epsilon(x : y :: z : \hat{t},\ x) = O_\epsilon(x : y :: z : t,\ x) - 1$ is obtained by applying the same permutations for all the intervals J such that $f_x(J) = \epsilon$ and $|\,J\,| = O$.

Finally, since

$$(f_x(i), f_t(i)) = (f_y(i), f_z(i)) \text{ with } f_x(i) = f_y(i) = \epsilon$$

and

$$(f_x(i+1), f_t(i+1)) = (f_z(i+1), f_y(i+1)) \text{ with } f_x(i+1) = f_z(i+1) = \epsilon,$$

we have

$$\begin{aligned} \max(|\,f_x(i-1)\,|, |\,f_t(i-1)\,|) + \max(|\,f_x(i)\,|, |\,f_t(i)\,|) & \\ + \max(|\,f_x(i+1)\,|, |\,f_t(i+1)\,|) & \\ = \max(|\,f_x(i-1)\,|, |\,f_t(i-1)\,|) + |\,f_z(i)\,| + |\,f_y(i+1)\,| & \\ \leq \max(|\,f_x(i-1)\,|, |\,f_t(i-1)\,| + |\,f_y(i+1)\,|) + |\,f_z(i)\,|, & \end{aligned}$$

which yields

$$T(x : y :: z : \hat{t}) \leq T(x : y :: z : t). \qquad \square$$

B Proof of proposition 12

In the following, we show that, for a tree t, the Algorithm 1 constructs an automaton A_t such that the successful paths in A_t are exactly the P-factorizations of t.

Let us first remark that the set of nodes $Pos(t)$ of a tree t is equipped with a natural partial order[22] $\leq$ defined as follows: $\forall (p_i, p_j) \in Pos(t)^2$, $p_i \leq p_j$ if and only if

[22] Trees can indeed be defined as particular partially ordered sets.

$\exists p \in \mathbb{N}^*$ such that $p_i = p_j.p$. This simply states that $p_i \leq p_j$ if and only if p_i is a descendant of p_j, i.e. p_j dominates p_i. An *antichain* with respect to this partial order is a subset $a \in Pos(t)$ in which each pair of elements is incomparable, that is: $\forall (x, y) \in a^2$, neither $x \leq y$ nor $y \leq x$ (no node is dominating another node). Given an antichain $a = \{p_1, \ldots, p_n\}$ of $Pos(t)$, replacing the subtrees rooted at $(p_1, \ldots, p_n)$ in t with the variables $(p_1, \ldots, p_n)$ results in a tree t' such that $\mathcal{VP}os(t') = a$.[23] Moreover, the application $\mathcal{VP}os$ is defined for any such t'; $\mathcal{VP}os$ is thus bijective. For readability, $\mathcal{VP}os$ is noted γ in the following.

Lemma 2. *A bijective correspondence can be established between any P-factorization $[(t_1, \ldots, t_n), (p_1, \ldots, p_{n-1})]$ of a tree t and the sequence of antichains $(a_1, \ldots, a_n)$ defined as $\forall i \in [\![1, n]\!]$, $a_i = \gamma(t_1 \lhd_{v_1} \ldots \lhd_{v_{i-1}} t_{i-1})$. This sequence verifies $a_n = \emptyset$, and $\forall i \in [\![1, n]\!]$, $a_{i+1} = a_i \setminus \{p_i\} \cup p_i.\gamma(t_i)$.*

Proof. Let $[(t_1, \ldots, t_n), (p_1, \ldots, p_{n-1})]$ be a P-factorization of t. We note for i in $[\![1, n]\!]$, $t'_i = t_1 \lhd_{v_1} \ldots \lhd_{v_{i-1}} t_{i-1}$. The tree t'_{i+1} is obtained by replacing in t'_i the variable p_i at position p_i with t_i. It is thus routine to verify that $\gamma(t'_{i+1}) = \gamma(t'_i) \setminus \{p_i\} \cup p_i.\gamma(t_i)$, and $\gamma(t'_n) = \gamma(t) = \emptyset$.

Conversely, let $(a_1, \ldots, a_n)$ be a sequence of antichains such that $a_n = \emptyset$, and for $i \in [\![1, n]\!]$, there exists a position p_i and a tree t_i with $a_{i+1} = a_i \setminus \{p_i\} \cup p_i.\gamma(t_i)$. For $i \in [\![1, n]\!]$, we set and $t'_i = \gamma^{-1}(a_i)$. We have $\forall i \in [\![1, n]\!]$, $t'_{i+1} = t'_i \lhd_{p_i} t_i$, and $t'_n = \gamma^{-1}(\emptyset) = t$, so $[(t_1, \ldots, t_n), (p_1, \ldots, p_{n-1})]$ defines a P-factorization of t. □

Lemma 3. *Let $A_t = \langle Q, E, i, f \rangle =$`ConstructAutomaton(t)`. There exists a bijection β from the set of states Q to the set of antichains over $Pos(t)$ such that $\beta(i) = \{\epsilon\}$, $\beta(f) = \emptyset$, and $(q_1, q_2, p_1 \to t_1) \in E$ if and only if $a_2 = a_1 \setminus \{p_1\} \cup p_1.\gamma(t_1)$, where $a_1 = \beta(q_1)$, $a_2 = \beta(q_2)$.*

Proof. By induction on the size of the tree. Let us consider the tree $t = l(t_1, \ldots, t_n)$ of size s_t. If $s_t = 1$, the only antichains of t are $a_1 = \{\epsilon\}$ and $a_2 = \emptyset$, which correspond exactly to the two states of the automaton constructed by `ConstructAutomaton`(t) (respectively i and f). The variable ϵ in $\gamma^{-1}(a_1)$ can rewrite either to itself or to l; these are the two transitions in E, which proves the result for $s_t = 1$ (see Lines 1-4).

We now assume the result is true for any tree of size $s < s_t$. Let a be an antichain of t; by definition of an antichain, either $a = \{\epsilon\}$ or $\epsilon \notin a$. The first case correspond to the initial state, added to the set of states (Line 5). In the second case, since $\epsilon \notin a$, a can be rewritten as $a = 1.a_1 \cup \cdots \cup n.a_n$, where a_i is an antichain of t_i for $i \in [\![1, n]\!]$. Conversely, if a_i is an antichain of t_i for $i \in [\![1, n]\!]$, $1.a_1 \cup \cdots \cup n.a_n$ defines an antichain of a such that $\epsilon \notin a$. By induction hypothesis, there exists a bijection between the antichains of $Pos(t_i)$ and the states of `ConstructAutomaton`(t_i). Consequently, there is also a bijection between the antichains of $Pos(t)$ and the states of `ConstructAutomaton`(t) (Lines 7-11).

A replacement in t of variable $i.p$ with a tree u at position $i.p$, with $i \in [\![1, n]\!]$, corresponds to a replacement in t_i of variable p with a tree u' at position p such that $u' = \mathrm{add\_pref}(i, u)$, and vice-versa, so by induction hypothesis, the required property over the transitions is verified (Lines 12-15). A replacement in an empty tree of the variable ϵ with a tree u corresponds to a transition from the initial state i to the state q such that $\gamma(u) = q$ (Lines 6 and 16), which ends the proof. □

[23] The so-called *frontier operation* in the Data-Oriented Parsing litterature.

Proof. Finally, to prove that the successful paths in A_t are exactly the P-factorizations of t, it is sufficient to note that the P-factorizations exactly correspond to the sequences of antichains that verify certain conditions (Lemma 1), and that these very sequences of antichains correspond to the sucessful paths in A_t (Lemma 2). □

C Implementation details

The general algebraic analogical framework presented in this paper has been partly implemented in a software package called ALANIS (A Learning-b ANalogy Inferencer for Structured data). The goal of this implementation is to make concrete the mathematical foundations by furnishing the algorithms needed to compute analogical proportions. Moreover, we want these algorithms to work on the large variety of structures covered by the formal model.

In order to achieve this goal of genericity, our implementation is based on the design-pattern underlying the Vaucanson library [Lombardy et al., 2003]. This design-pattern has first been conceived to manipulate generalized automata, i.e. automata whose transitions may be labeled by letters, but also words or series [Sakarovitch, 2003]. The great particularity of this design-pattern is to completely separate the algebraic structures (e.g. alphabets, monoids, semirings and series) from their implementations (e.g. characters, strings, vectors or maps). This genericity heavily relies on static-linking techniques which ensure that the framework does not suffer from runtime overhead [Régis-Gianas and Poss, 2003].

This design-pattern forms the core of our framework: the algebraic structures (which give a meaning to analogical proportions) and their concrete implementations are totally separated. For example $float$ or $double$ may be used to modelize elements of $(\mathbb{R}, +)$ or $(\mathbb{R}^*, \times)$ and $integers$ may be used to modelize elements of $(\mathbb{Z}, +)$. In every case, the underlying algebraic structure is an abelian group and there is a single algorithm to handle all these situations. Moreover, it is possible to specialize an algorithm with respect to its implementation: additional optimizations can be performed when the underlying implementation of a data structure has some features we can take advantage of while preserving the framework core.

As a consequence, our framework is notably modular and it is very easy to add new types of objects if the corresponding underlying algebraic structure is already handled.

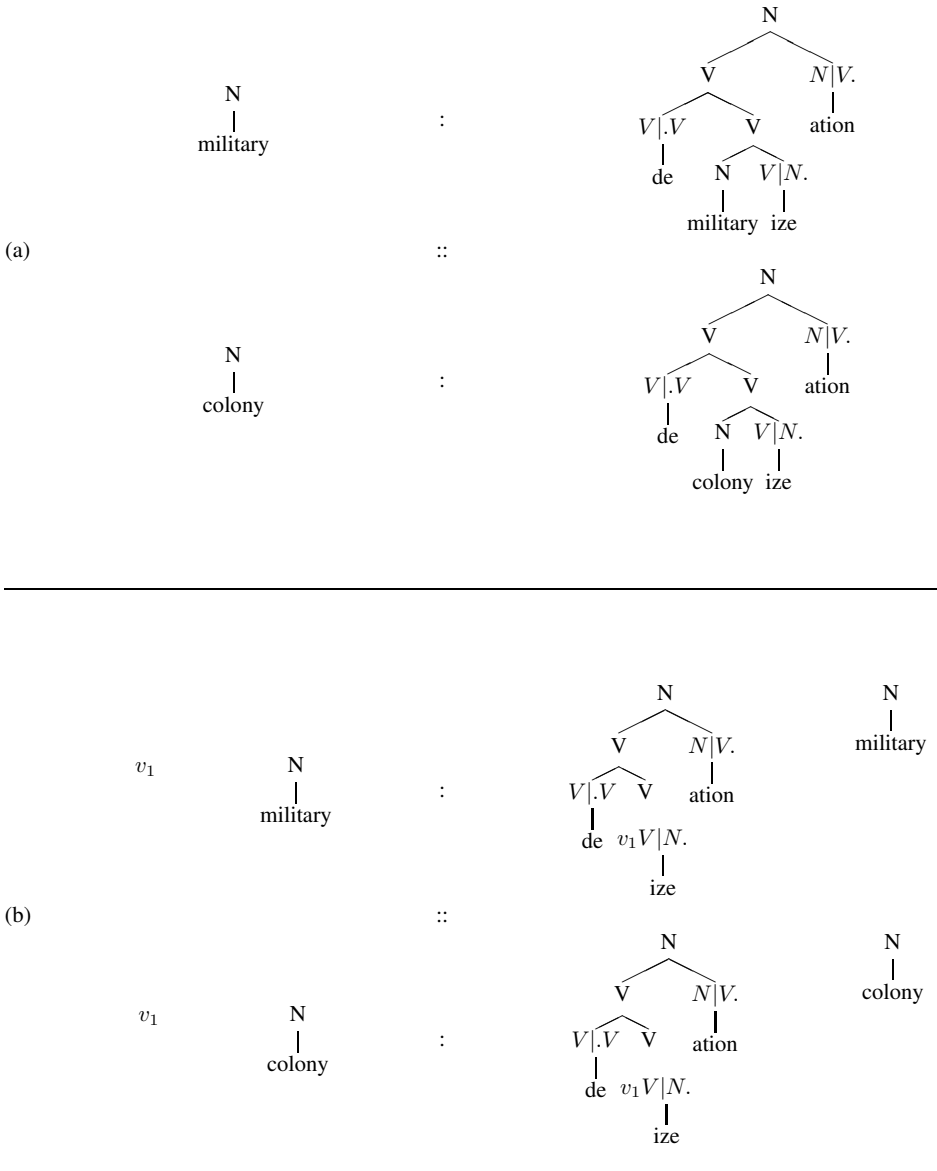


Figure 17: $military$: $demilitarization$ :: $colony$: $decolonization$ (with implicit (a) and explicit (b) factorizations)

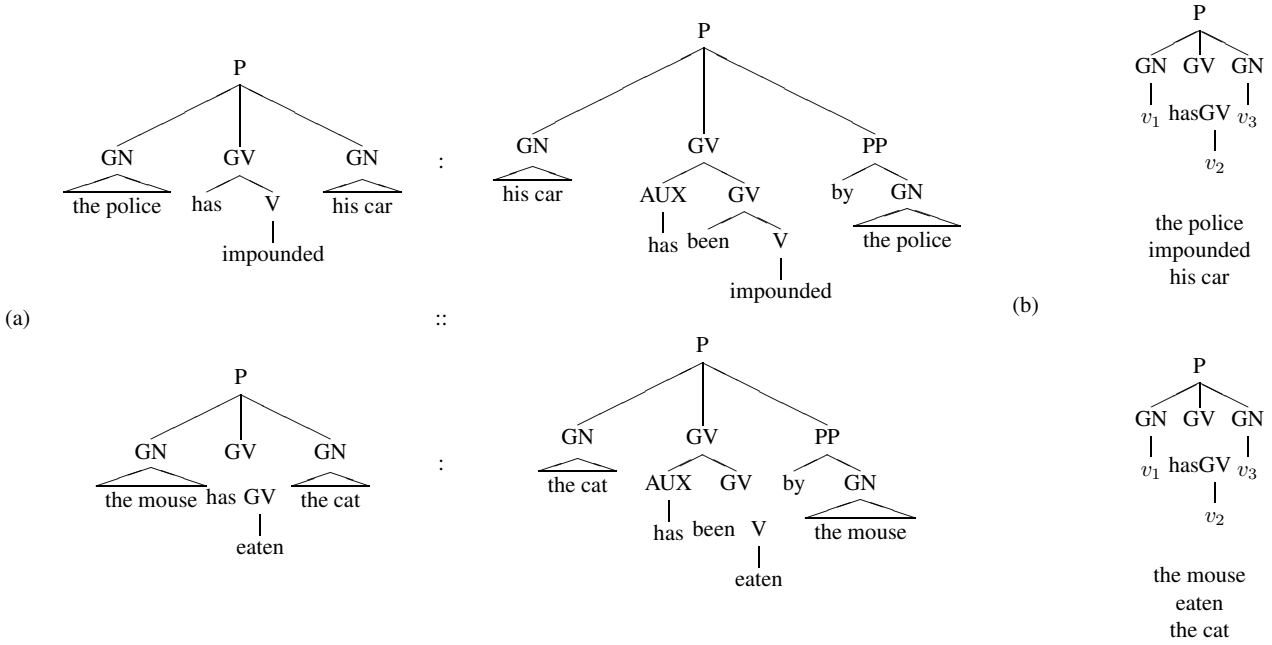


Figure 18: Active/passive proportion (without (a) and with (b) factorizations)

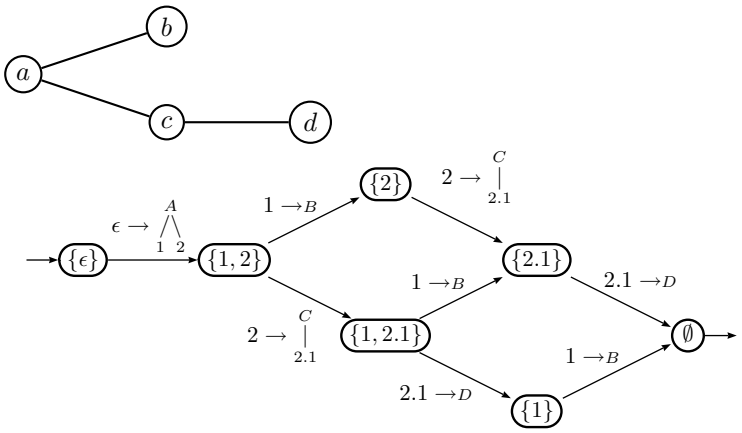


Figure 19: From tree to automata. The transitive closure and ϵ loops have been omitted for clarity.