Analogies Between Binary Images: Application to Chinese Characters

Yves Lepage

Abstract The purpose of this chapter is to show how it is possible to efficiently extract the structure of a set of objects by use of the notion of proportional analogy. As a proportional analogy involves four objects, the very naïve approach to the problem, has basically a complexity of $O(n^4)$ for a given set of n objects. We show, under some conditions on proportional analogy, how to reduce this complexity to $O(n^2)$ by considering an equivalent problem, that of enumerating analogical clusters that are informative and not redundant. We further show how some improvements make the task tractable. We illustrate our technique with a task related with natural language processing, that of clustering Chinese characters. In this way, we re-discover the graphical structure of these characters.

1 Introduction

Proportional analogy is defined in various ways by different recent authors [1–3]. Referring back to the most ancient definitions, one can reach an agreement on the following definition:

> Four objects, A, B, C and D, form a proportional analogy if the first object is to the second object in the same way as the third object is to the fourth object. A proportional analogy is noted $A : B :: C : D$.

In all generality, if the relation between two objects (noted by the colon :) is called a *ratio* and the relation between the two pairs of objects (noted by the two colons ::) is called a *conformity,* then a proportional analogy is a conformity of ratios between two pairs of objects.

Y. Lepage (✉)
Graduate School of Information, Production and Systems, Waseda University,
808-0135 Hibikino 2-7, Wakamatsu-ku, Kitakyushu-shi, Fukuoka-ken, Japan
e-mail: yves.lepage@waseda.jp
URL: http://www.waseda.jp/ips/english/kyoin/01_8.html

H. Prade and G. Richard (eds.), *Computational Approaches to Analogical Reasoning: Current Trends*, Studies in Computational Intelligence 548,
DOI: 10.1007/978-3-642-54516-0_2,

Table 1 Examples of proportional analogies between words

Proportional analogy	Levels on which the analogy holds	
	Form	Meaning
to walk : *walked* :: *to work* : *worked*	Yes	Yes
wings : *fins* :: *a wing* : *a fin*	Yes	Yes
to walk : *walked* :: *to be* : *was*	No	Yes
wings : *fins* :: *bird* : *fish*	No	Yes
to walk : *walked* :: *to me* : *meed*	Yes	No
wings : *fins* :: *winged* : *fined*	Yes	No

Proportional analogy can be seen between words on the level of form or on the level of meaning or on both at the same time (see [4] for abnormal cases). Table 1 gives examples of analogies that hold on these two levels at the same time or only on one of these two levels.

Proportional analogies on the levels of form and meaning at the same time are called true analogies. Between chunks or short sentences, their number has been shown to be quite important [5–7]. Many studies, too many to cite here, address the efficiency of analogy for segmenting words or grouping them according to word families (as for Chinese, see for instance [8]). Forms which depart from declension or conjugation paradigms (groups of proportions in [9]) were called anomalies in Classical grammar [10]. Recently, analogies between word meanings (*water* : *riverbed* :: *traffic* : *road*) have been shown to be reproduceable on computers using large corpora and vector space models [11–13].

1.1 Proportional Analogies Between Binary Images

Proportional analogies are not only verbal. They may hold between any kind of objects provided, generally, that the objects be of the same kind, a point Aristotle, among other ancient and recent authors, insists on. The general principle, viewed as a cognitive process, is based on iconicity [14]. Taking the term to its restrained graphical sense, the example in Fig. 1 illustrates a proportional analogy between binary images made of black and white pixels.

This analogy makes sense for human beings. Two oppositions are seen in this analogy. The first opposition is between a happy and a sad expression. The second opposition is between two faces (of two people). The fact that the two faces may be interpreted as belonging to a male and a female character may depend on cultural, social or even individual judgements.

1.2 Proportional Analogies Between Chinese Characters

The example in Fig. 1 is a purely graphical example. The example in Fig. 2 is also a graphical proportional analogy but the images represent Chinese characters

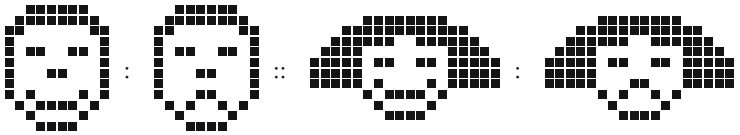

Fig. 1 An analogy between binary images made of *black and white* pixels

written in some given font. As is true with the analogy in Fig. 1, this analogy is understandable by anybody, i.e., the oppositions expressed by the analogy can be extracted by anybody, even people without a prior knowledge of the Chinese writing system. Indeed, it is understandable that the left parts (not printable in isolation) and the right parts (隹, 吉) of the characters can be exchanged to give rise to the four different characters present in the analogy: 维, 结, 谁 and 诘. In the same way as with the but last example of Table 1, *to walk* : *walked* :: *to me* : *meed*, this is only an analogy of form. Indeed, this analogy does not apply on the level of meaning, as the meanings of these characters are unrelated: 'maintain, preserve', 'knot, tie', 'who' and 'question, interrogate'. It does not constitute an analogy on the level of pronunciation either: /wéi/, /jié/, /shuí/ and /jié/.

In [15], where the goal is to compute similarity between Chinese characters, this kind of decomposition of characters into elementary parts is performed thanks to prior knowledge about the Chinese writing system, so that the elements into which the decomposition is performed are not accessible to anybody, let alone a machine that would only be given the binary images as input.

The particular and practical problem which we tackle in a broader research concerned with the ease of learning the Chinese characters for human beings [16], is to re-discover the graphical structure of Chinese characters in a fully automatic way by relying on the notion of proportional analogy, without any prior knowledge of the Chinese writing system. This graphical structure can be made explicit by finding analogies between characters, as exemplified by Fig. 2. In a first step, we are only concerned with gathering such graphical analogies, possibly, all the possible ones. Intersecting graphical analogies with analogies on the levels of pronunciation and meaning is performed only in subsequent steps of our research not reported in this chapter.

The problem tackled in the present chapter is thus only that of recognizing graphical analogies between binary images made of black and white pixels, but our data will be Chinese or Sino-Japanese characters. In other terms, the general and theoretical problem that we tackle in the following sections is to rely on the properties of proportional analogies so as to automatically visualize the structure of a set of objects, binary images in general, Chinese characters in particular.

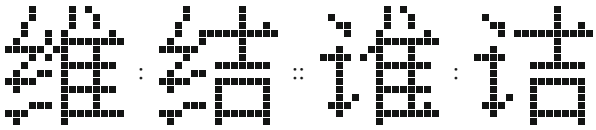

Fig. 2 Another analogy between binary images made of *black and white* pixels

1.3 Structure of the Chapter

The chapter is structured as follows: Sect. 2 introduces the general problem of enumerating all analogies between objects contained in a set of objects and its complexity. Section 3 shows how the problem can be transformed into a problem of quadratic complexity and introduces the notion of analogical clusters for this purpose. Section 4 gives our proposed method to output analogical clusters. Section 5 mentions some improvements that can reduce computational time. Section 6 describes the application of the proposed method to the problem of structuring Chinese characters, and describes the results obtained in our experiments.

2 Enumerating All Analogies Between Objects in a Set of Objects

The naïve approach to the general problem of the enumeration of all proportional analogies between a set of n objects consists in examining all possible quadruples of objects and checking for analogy. This naïve approach has a complexity of $O(n^4)$.

Without changing the complexity, the computation time may be reduced. For a given proportional analogy, there exists seven other equivalent forms (see Theorem 2.1 in [17]). This is implied by the basic properties of exchange of the means (exchanging objects B and C in the analogy $A : B :: C : D$, second line below) and symmetry of conformity (exchanging the terms on both sides of the :: sign, sixth line below). By applying these basic properties in any order iteratively, the following eight analogies are shown to be equivalent:

$A : B :: C : D$
$A : C :: B : D$ exch. means
$B : A :: D : C$ exch. means + sym. :: + exch. means
$B : D :: A : C$ exch. means + sym. ::
$C : A :: D : B$ sym. :: + exch. means
$C : D :: A : B$ sym. ::
$D : B :: C : A$ sym. :: + exch. means + sym. ::
$D : C :: B : A$ exch. means + sym. :: + exch. means + sym. ::

Because of these eight equivalent forms, the enumeration time can be divided by a factor of 8, but the complexity remains $O(n^4)$.

To make our point clear, consider the following naïve estimation. In a preliminary experiment, we estimated the average time needed for to verify one analogy between four Chinese characters using 36 features (see Sect. 6.3 for a description of the features). An average time of 0.8 ms was measured. For almost five thousand Chinese characters (see Sect. 6.3 for a description of the data), and knowing that there are a little bit less than 3.2×10^7 s in a year, verifying all possible analogies would take almost 400 years.[1]

In order to reduce the complexity of this problem, we propose to modify our goal. Rather than aiming at individual analogies, we compute all possible ratios between all possible objects at hand. This computation is basically $O(n^2)$. The result of this computation allows us to cluster pairs of objects according to their ratios. These clusters summarize all possible analogies between all objects in a non-redundant way that still provides the total amount of information (see Sect. 3). The sequel of the chapter shows how to compute such clusters and presents some of the actual results of such a computation on a set of Chinese characters.

3 Analogical Clusters

3.1 Objects as Feature Vectors

In this work, we represent an object by a vector of features with numerical values. We also impose that the feature space be the same for all objects, so that it is trivially possible to define a ratio between two objects as the vector of their difference. In such a setting, conformity is trivially reduced to equality between vectors, more precisely equality between difference vectors.

The following equation illustrates a possible case of proportional analogy between vectors in a four-dimensional space.

$$\begin{pmatrix}1\\0\\0\\1\\0\end{pmatrix} - \begin{pmatrix}0\\0\\0\\0\\0\end{pmatrix} = \begin{pmatrix}3\\2\\2\\1\\2\end{pmatrix} - \begin{pmatrix}2\\2\\2\\0\\2\end{pmatrix}$$

[1]
$$\begin{aligned} 5{,}000^4 \times 0.8\ \text{ms}/8 &> 5^4 \times 10^9 \times 0.1\ \text{s} \\ &> 125 \times 10^8\ \text{s} \\ &> 1250 \times 10^7/(3.1563 \times 10^7)\ \text{years} \\ &> 394.2\ \text{years}. \end{aligned}$$

Vector difference as a ratio, and equality between vectors as conformity, consistently define analogies that meet the intuitive notions about proportional analogies. Among other properties, the eight forms of equivalence for the same proportional analogy (see Sect. 2) always hold.

3.2 Transitivity of Conformity: Analogical Clusters

It is not always the case that conformity verifies transitivity. For instance, [18] shows that the intuitive notions of proportional analogy between strings of characters imply that there is no transitivity for conformity.[2]

In our setting with conformity being an equality, i.e., an equivalence relation, transitivity naturally holds in addition to reflexivity and symmetry. For proportional analogies, transitivity of conformity implies that:

$$A : B :: C : D \quad \text{and} \quad C : D :: E : F \Rightarrow A : B :: E : F$$

For our present task of enumerating all possible proportional analogies between all objects in a given set, a transitive conformity can lead to an enormous economy in representation. To illustrate this point, consider the following three proportional analogies.

$$\begin{array}{l} A : B :: C : D \\ C : D :: E : F \\ A : B :: E : F \end{array}$$

They can be represented in a more economical way by the following list of equal ratios:

$$\begin{array}{l} A : B \\ C : D \\ E : F \end{array}$$

All ratios being equal, any possible proportional analogy formed by taking any two ratios holds.

From the above example, it is clear that, provided conformity is transitive, a list of n pairs of objects with the same ratio stands for a list of $n \times (n-1)/2$ non-trivial proportional analogies (see Sect. 3.5 for trivial analogies). Consequently, under the assumption of transitivity for conformity, the problem of enumerating all possible proportional analogies between all possible objects in a given set can be transformed into a problem of enumerating all possible pairs of objects with the same ratio. The

[2] This comes from the fact that some analogies between strings of characters admit multiple solutions. When this is the case, then, there is not transitivity for :: in the general case for the objects considered (see [18, p. 113]).

former problem has a complexity of $O(n^4)$ while the latter one has a complexity of $O(n^2)$.

From now on, we shall call a list of ratios of objects with the same value, an *analogical cluster.*

3.3 Equivalent Forms of Analogy: Redundancy of Clusters

Each analogical cluster stands for a different ratio, i.e., a vector that represents the difference between any two feature vectors each representing an object.

Because the order in analogical clusters is not relevant, an analogy extracted from an analogical cluster stands for two equivalent forms, obtained by symmetry of conformity (sixth line in the eight equivalent forms of proposition analogy in Sect. 2).

$$A : B :: C : D \quad \Leftrightarrow \quad C : D :: A : B$$

By inversion of ratios (third line in the eight equivalent forms of proportional analogy in Sect. 2), a proportional analogy involves two different ratios. And by exchange of the means, (second line in the eight equivalent forms of proposition analogy in Sect. 2), another two different ratios.

$$A : B :: C : D \quad \Leftrightarrow \quad B : A :: D : C \quad \Leftrightarrow \quad A : C :: B : D$$

Consequently, in total, the eight different forms of the same proportional analogy are to be found in four different analogical clusters (and only four clusters) among all the possible clusters that are output by a method yielding all the possible clusters standing for differences between all feature vectors representing all the objects in a given set.

Figure 3 shows such four analogical clusters for the proportional analogy $A : B :: C : D$. These four clusters are redundant because of the eight equivalent forms for the same proportional analogy, as we have just stated:

- In any cluster, the order of appearance of the pairs of objects being irrelevant, each cluster encapsulates two equivalent forms of the same proportional analogy. This is symmetry of conformity.
- Analogical clusters (1) and (2) together contain the same information as clusters (3) and (4) together. The relation between (1) and (2) (and between (3) and (4)) is the exchange of the means.
- Analogical clusters (1) and (3) are indeed the same up to an exchange of the objects on the left and the right of the : sign. This is actually inversion of ratios. The same is also true for clusters (2) and (4).

It is trivially possible to eliminate the redundancy between clusters (1) and (2) and clusters (3) and (4). This can be done by avoiding the computation of the difference

Cluster number			
(1)	(2)	(3)	(4)
$A : B$	⋮	⋮	⋮
⋮	$B : D$	$B : A$	$C : A$
⋮	⋮	⋮	⋮
$C : D$	$A : C$	⋮	$D : B$
⋮	⋮	$D : C$	⋮

Fig. 3 For a given proportional analogy $A : B :: C : D$, the set of analogical clusters output by a method that looks for all possible vector differences between all possible feature vectors representing objects in a given set should include four clusters

between two vectors and its opposite value (the same two vectors in the reverse order). For that, it suffices to sort all the vectors in some predefined increasing order, and to compute only the differences between two vectors ranked in that order. In this way, a particular proportional analogy will appear in two, and only two, different analogical clusters among the set of all clusters. As a result, globally, the set of all clusters will contain no redundant information.

3.4 Equality of Feature Vectors: Separation of Space

By definition of the ratio as a vector difference, the case where $A : B$ and $A : C$ belong to the same analogical cluster is only possible if the vectors representing B and C are the same. This can only happen if the feature vectors do not separate the space of objects into each individual object. Reciprocally, if the feature vectors are unique for each different objects in the given set, the two ratios $A : B$ and $A : C$ for different B and C will be different. For our proposed method, this implies to check for the *separation of the space of objects* before proceeding to clustering.

3.5 Trivial Analogies: Informativity of Clusters

Finally, we must mention a particular case of no interest as it does not bring any information. This is the special case of the cluster for the null vector; i.e., null ratio. It has the following form.

$$\begin{array}{c} A : A \\ B : B \\ C : C \\ \vdots \end{array}$$

It represents the set of all *trivial proportional analogies,* i.e., proportional analogies of the form: $A : A :: B : B$. As our interest is the enumeration of informative analogical clusters we simply avoid to produce this cluster.

By exchange of the means, trivial analogies are equivalent to analogies of the form $A : B :: A : B$. Enumeration of pairs of objects in a predefined sorting order trivially ensures that the difference between two objects is never computed twice. However, it does not prevent from outputting clusters that would contain only one pair of objects. This happens when two objects have a unique vector difference. This problem will be tackled in Sect. 5.1.

4 Informative and Non-redundant Enumeration of Analogical Clusters

4.1 Feature Tree and Quadratic Exploration of the Feature Tree

Each object is represented by a vector of features. An order can be imposed on the features. In this way, each vector is considered as a list with a recursive structure of a head (the first feature value) and a tail (the remaining features). The lexicographic order, relying on the order on integers, can be applied to such a set of lists. In this way, it is possible to sort the feature vectors representing all objects.

A tree structure underlies such an ordered list. For the first feature, each different value can be encoded in one node. Each such node can be assigned the interval that represents the span over the sorted list of objects. This can be recursively applied to each interval with the tail of the feature vectors considered as lists (thus for the second feature and so on) to build a tree structure where the levels stand for each different feature and where each node holds the interval of the sorted objects with the same value for that feature, all values above being equal. On the last level, each interval should be reduced to one object if the space is well separated. Such a tree structure can be traversed in breadth-first order. Figure 4b illustrates such a data structure for a set of 5 feature vectors given in Fig. 4a.

This tree structure is the same as the one we used in two of our previous works: for the complete enumeration of all analogies between sentences contained in corpora of 100,000 short sentences in Chinese, Japanese and English [5] (with sequences of bits as features and various ratios for various features and automatic sorting of the features for early detection of useless zones in the cluster space so as to speed up the overall process); and for the enumeration of clusters reflecting linguistic oppositions among 40,000 short sentences in English and Japanese [19]. In these two works, respecting the equality of edit distance for analogies of commutation between strings of characters implied extra processing.

This tree structure is quite different from the one used in [20] to search a space of strings of characters for analogies. Most importantly, our goal is different as we aim at a complete enumeration of all possible ratios, which compelled the design of the

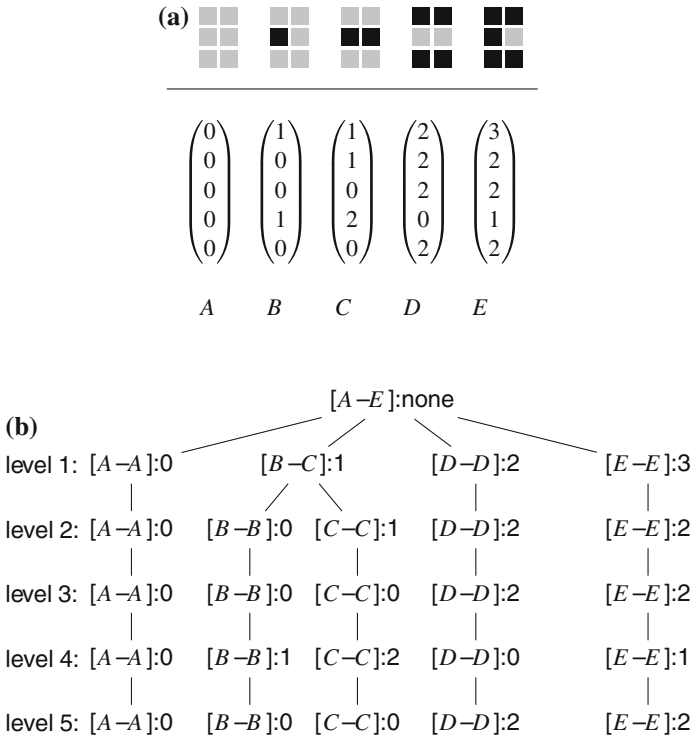

Fig. 4 A set of objects represented as feature vectors and its corresponding feature tree. **a** A set of five objects. Each object is a binary image of size 2 columns × 3 rows. For each object a feature vector of size 5 can be built by counting the number of *black* pixels on each column and on each row. The objects are sorted according to a given sorting procedure, here, the lexicographic order based on the natural order on numbers; vectors are read *top-down*. One analogy can be seen among these objects: : :: :, i.e., $A : B :: D : E$. This analogy is equivalent to the analogy between vectors given in Sect. 3.1. **b** The feature tree corresponding to the set of five objects represented as feature vectors given in Fig. 4a. The intervals, noted with brackets (e.g., $[B–C]$), are followed by the value of the feature on that level (e.g., 1 for $[B–C]$). The letters in the intervals refer to the naming of the objects given at the *bottom* of Fig. 4a

proposed data structure. In [20], the goal is more specific as it consists in retrieving all the possible pairs of objects that would be in analogical relation with a given pair of objects. This implies the two main differences given hereafter.

First, the geometry is different. In [20], the nodes on the same level may correspond to different characters (i.e. features). This is not the case here. Each level must correspond to exactly the same feature. Consequently, on the contrary to the structure in [20] no intermediate node can stand for an object. All the objects are to be found on the leaves.

Second, the labels borne by the nodes are different. In our proposed data structure, the nodes bear the spanning interval in the sorted list of objects and the value of

the feature; the name of the feature, being useless, is forgotten. This is of primary importance for the parallel traversal in sorted order of objects with the first interval never overtaking the second interval so as to avoid redundancy (see Sect. 3.3 and see below).

The computation of all ratios between all feature vectors simply consists in traversing the same tree in parallel in breadth-first order (a kind of a Cartesian self-product), and computing the differences between the values on each pair of nodes. For the same difference at a given local level, all the pairs of intervals are memorized as a block. This procedure is recursively applied down to the last level for each different value at a local level. Figure 5 illustrates this process for the feature tree given in Fig. 4.

4.2 Sketch of the Method

The following gives a sketch of the proposed method.

- Convert each object into a feature vector;
- check for separation of space;
- define an order on the feature vectors (we use least correlations of values among features);
- sort the feature vectors according to lexicographic order in the defined order of features;
- build a feature tree for the sorted feature vectors;
- traverse the feature tree in parallel in breadth-first order to compute the differences between the feature vectors by blocks;
- output the list of pairs of intervals (on the last level, each interval should be reduced to one object if the space is well separated) that corresponds to each vector difference.

By construction and by definition, each list of pairs of objects, that share the same feature vector difference, is an analogical cluster.

The computation in Fig. 5 already illustrates the following improvements that can be made to the computation.

Sections 5.1 and 5.2 show that it is possible to terminate the exploration by checking for some structural conditions on the list of pairs of intervals memorized.

In the parallel traversal, we impose that for two lists of pairs of intervals to be processed, the first list be strictly before the second list. This is tantamount to explore only the upper corner of a matrix excluding its diagonal. This avoids redundancy and non-informativity when computing all possible analogies: the ratio of two vectors is computed once, its opposite is not (Sect. 3.3); intervals that are reduced to one object on the diagonal are checked to avoid trivial clusters (Sect. 3.5).

The effect of these improvements is an important practical reduction in computation. For the example in Fig. 5, the number of cells computed to output the only analogy that can be found in the set of objects is:

(a)	[*A* –*A*]:0	[*B* –*C*]:1	[*D* –*D*]:2	[*E* –*E*]:3
[*A* –*A*]:0		1	2	3
[*B* –*C*]:1		0	1	2
[*D* –*D*]:2				1
[*E* –*E*]:3				

(b)	[*A* –*A*]:0	[*B* –*B*]:0	[*C* –*C*]:1	[*D* –*D*]:2	[*E* –*E*]:2
[*A* –*A*]:0		1,0	1,1	2,2	
[*B* –*B*]:0			0,1	1,2	2,2
[*C* –*C*]:1				1,1	2,1
[*D* –*D*]:2					1,0
[*E* –*E*]:2					

(c)	[*A* –*A*]:0	[*B* –*B*]:0	[*C* –*C*]:0	[*D* –*D*]:2	[*E* –*E*]:2
[*A* –*A*]:0		1,0,0	1,1,0	2,2,2	
[*B* –*B*]:0					2,2,2
[*C* –*C*]:0				1,1,2	
[*D* –*D*]:2					1,0,0
[*E* –*E*]:2					

Fig. 5 Illustration of the computation of all ratios for the set of objects given in Fig. 4a. In each cell, the value inherited from the previous level is concatenated with the difference: value in column minus value in row on the current level. Empty *white* cells are not computed either to avoid redundancy (opposite values) or because they correspond to a ratio that is unique. *Black* cells are not considered because they would give rise to trivial analogies: they are located on the diagonal and their corresponding interval is reduced to one object. **a** Computation of the value differences on the first level for the feature tree of Fig. 4b (refer to that figure for the values on this level). The blocks with the same values are the seeds of possible clusters of same ratio. **b** Recursive computation of the value differences on the second level (refer to Fig. 4b for the values on the second level; the first numbers before the comma in the cells are inherited from the first level in Fig. 5a). The blocks with the same values on the first level are further decomposed into sub-blocks. For example, the block in Fig. 5a with value 1 (3 cells) gives birth to two sub-blocks, one with value of 1, 0 and the other one with value 1, 1. All other blocks with different values will not be worth further exploring as they all consist of one cell of intervals reducing to one object. **c** Recursive computation of the value differences on the third level (same principle as Fig. 5b). The block with value 1, 0, 0 corresponds to the two cells $[A–A] \times [B–B]$ and $[D–D] \times [E–E]$; further exploration on the next lower levels will check for equality of differences of values until the last level; thus, these two cells represent the analogy $A : B :: D : E$. The same thing happens for the two cells $[A–A] \times [D–D]$ and $[B–B] \times [E–E]$ standing for the analogy $A : D :: B : E$ (exchange of the means of the previous one). The two cells with values 1, 1, 0 and 1, 1, 2 will not be further explored as they consist of one cell of intervals reducing to one object; they would give rise to degenerated clusters

$$
\begin{array}{rl}
 & 7 \text{ cells on the 1st level} \\
+ & 9 \text{ cells on the 2nd level} \\
+ & 6 \text{ cells on the 3rd level} \\
+ & 4 \text{ cells on the 4th level} \\
+ & 4 \text{ cells on the 5th level} \\
\hline
= & 30 \text{ cells}
\end{array}
$$

out of all the possible 125 cells (5 objects $\times$ 5 objects $\times$ 5 levels). This is a reduction of 75 % of the computation.

5 Improvements

5.1 Elimination of Clusters Reduced to One Pair of Objects

We call degenerated clusters those clusters which contain only one pair of objects, $A : B$. Obviously, such clusters do not give rise to any analogy other than the trivial analogy $A : B :: A : B$ and are thus not worth to output. An early detection of such cases leads to an important reduction in processing time.

The implementation of the early detection of such degenerated clusters relies on the data structure of feature tree. After the computation of all possible differences between all possible vectors down to a certain level in the tree, it is easy to scan all the differences and look at the intervals they represent. If a set of pairs of intervals contains only one pair of intervals, each of which being reduced to one object, this is a case of a degenerated cluster. Such a cluster may be immediately deleted so as to stop any further computation on the lower levels.

A comparison of two runs of the programs with or without early detection of degenerated clusters is given in Table 2. It shows that, for our special case of structuring Sino-Japanese characters, a reduction of one third of the computational time can be achieved. There exists some overhead as is shown by the fact that an increase of 55 % in computational time is observed for 1,000 characters.

5.2 Conditional Elimination of Clusters Reduced to One Proportional Analogy

In Sect. 3.5, it was shown that an analogy appears in only two analogical clusters. For economy of description, we would like to eliminate redundant information as most as possible. When an analogy belongs to two clusters that contain a large number of pairs of objects, it is *a priori* impossible, without loss of information, to remove those lines that correspond to this analogy from one of the cluster. This is not the case when one of the analogy is reduced to a cluster that contains only one analogy,

Table 2 Comparison of runtimes with or without detection of degenerated clusters (clusters reduced to one pair of objects). The computation runtimes without detection of degenerated clusters are obtained for the technique described in Sects. 4.1 and 4.2

Number of chars Processed	Runtimes in seconds		Time reduction (%)
	Without	With	
1,000	9	14	+55
2,000	39	36	−7
3,000	92	82	−10
4,000	173	142	−17
5,000	277	219	−20
6,000	426	313	−26
7,000	605	438	−27
8,000	739	557	−24
9,000	944	702	−25
10,000	1204	836	−30
11,000	1517	1123	−25
12,000	1864	1302	−30
13,000	2265	1342	−40
14,000	2646	1791	−32
14,655	2873	1889	−34

i.e., exactly those two lines corresponding to the analogy at hand. This situation is illustrated below:

$$\begin{array}{l} \vdots \\ A : B \\ \vdots \\ E : F \\ \vdots \\ C : D \\ \vdots \end{array} \qquad \begin{array}{l} \\ \\ \\ A : C \\ B : D \\ \\ \end{array}$$

Here the analogy $A : B :: C : D$ can be read in two clusters. One on the left, under the form $A : B :: C : D$, and another one on the right, under the equivalent form $A : C :: B : D$. On the right, the cluster does not contain any other pair of objects than the two ones constituting the analogy in question. On the left, the two pairs $A : B$ and $C : D$ are just two pairs among many other pairs (noted by enumeration dots).

In such a case, the cluster on the right does not bring any additional information, as an equivalent form of the analogy is already present in the cluster on the left. Consequently, it is possible to delete the cluster reduced to one analogy, on the left, without deleting any information.

This can be performed during the enumeration of analogical clusters, level by level, using the feature tree. In this case, clusters reduced to one analogy should be memorized on each level. At the end of the exploration of each level of the feature tree, such clusters can be removed from the list of clusters to explore further. This

should lead to a reduction in the total computational time. Our current implementation does not make use of this possibility and performs the deletion of clusters reduced to one analogy after complete enumeration of analogical clusters, in a post-processing phase, which induces some additional computation cost.

6 Experiments

In the frame of a larger study concerned with measuring the ease with which learners can remember Chinese characters along with their pronunciation, we are interested in studying the regularities and the correspondences between the Chinese graphical forms of characters and their pronunciation.

It is known that Chinese characters exhibit some structure and are made of graphical elements which reflect either some iconic meaning or some pronunciation. As a first step in this study, we extracted all the possible analogies between Chinese characters using a fixed-sized font. We report hereafter some of the results obtained.

6.1 The Structure of Chinese Characters

Based on a classification given in Xǔ Shèn's book 說文解字/shuōwén jiězì/ 'Explaining and Analyzing Characters', written in the second century, the tradition considers six main categories (六書/lìushū/) for the constitution of individual Chinese characters:

- pictograms (象形): representation of the object itself, e.g. 月 'moon';
- symbols (指事): abstract representation of the notion, e.g. 上 'above';
- radical-pronunciation (形聲): combination of a meaning part and a pronunciation part, e.g. 河 'river', pronounced /hè/, is written as 可, the pronunciation of which is similar: /kè/, with the addition of a semantic graphical part for the notion of water: 氵;
- composition (會意): e.g. 林 'woods', 森 'forest' are obtained by reduplication of 木 'tree';
- modification (轉注): a character is adopted for another word with a similar pronunciation and meaning, e.g. 長 'long' (adj.) or 'to grow' (verb) depending on tone;
- borrowing (假借): a character used by extension to write another word with a similar pronunciation is specialized for this other word usuallybecause of a higher

京:先	identical left part (semantic key)	
凉:洗	氵	[water]
凉:冼	冫	[ice]
倞:侁	亻	[human]

氵:亻	identical right part (pronunciation clue)	
泮:伴	半	pàn
凉:倞	京	liàng
冼:侁	先	ēn

Fig. 6 On the *left*, on each line, the characters share the same semantic key. On the *right*, the characters share a same pronunciation indicated by the *right* part. Only four of the above characters are in use nowadays. The pronunciation clue refers to old pronunciations for some of these characters and is not necessarily valid for modern days Chinese pronunciation

frequency; the older word is then written with another character; e.g. the character for 'leather clothing', 求, being also used more and more for 'to seek' (same pronunciation), was replaced by 裘 = same character + 衣 'clothing'.

A large number of Chinese characters are obviously decomposable into components and it is usually claimed that almost 80 % of the Chinese characters can be considered as belonging to the 3rd category of radical-pronunciation characters. The most frequent structure of such characters consists of two elements, one being a pronunciation clue and the second one being a meaning clue, usually called semantic key, hence the name of phono-semantic characters. An illustration is given in Fig. 6.

This structure, although being quite common, is not valid for all characters. It is also believed that, because of phonetic changes, many characters that exhibited such kind of structure in ancient times cannot be interpreted in this way anymore.

In this chapter, we are not interested with the relationship between graphical form and pronunciation. Our goal is limited to the extraction of the graphical structure of Chinese characters by automatic means.

To perform our experiments, we use two sets of characters. The first one is a set of almost 15,000 Sino-Japanese characters and the second one is a set of 5,000 Chinese simplified characters. To compute the analogical structure of both sets, we use monospace (or fixed-width or fixed-size) fonts. Monospace fonts are lists of characters described as binary images of fixed height and width.

6.2 15,000 Sino-Japanese Characters

The font we use in our first experiment is knj10B.bdf.[3] It contains 14,655 Sino-Japanese characters in the range between the Unicode codepoints 19,968 (一) and 40,869 (龥). Figure 7 shows a sample of these characters. As shown in this figure, the characters in this font have a fixed height of 18 lines and a fixed width of 24 pixels. The actual width used is 18 pixels, so thatthis font is a 18 × 18 pixel font, the

[3] Font designed by Nagao Sadakazu (snagao@tkb.att.ne.jp). We use version 1.1 of 1999.

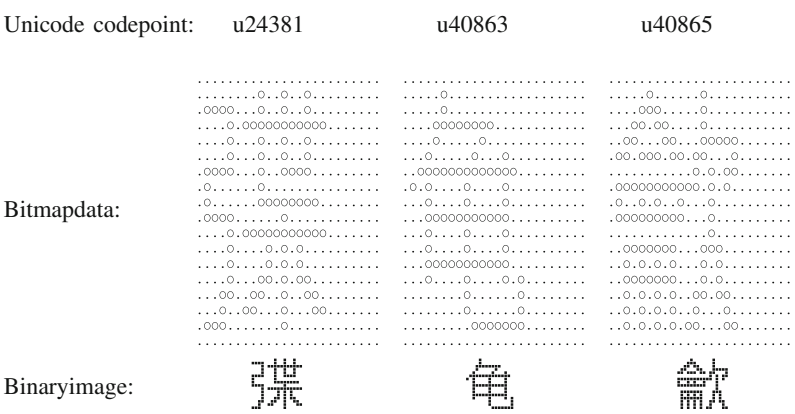

Fig. 7 Visualization of three Sino-Japanese characters (___, 齾 and 隲) from the font knj10B as binary images. The second one is not used in Japanese but belongs to the font. *White* pixels are visualized as a *dot*, *black* pixels as a small *circle*

value of 24 comes from the encoding of each line of the icons on 3 bytes that we use without change. Figure 7 visualizes the representation of three randomly selected characters from this font.

Features Used The features we used are simply the number of black pixels on each row and each column. We use 18 lines and 18 columns; making up a sum of 36 features per character. As an illustration, the feature vector for the leftmost character in Fig. 7 is given in Fig. 8.

We checked that the space of objects is completely separated by these features, i.e., each vector of feature values represents only one object in the set of objects. The above 36 features discriminate each character among the 14,655 characters used in our experiments.

Analogical Clusters Obtained We applied out method to extract the graphical structure of the 14,655 characters in our selected fixed-point font. The program, written in Python, needed less than 30 min to terminate.[4]

Figure 9 shows a sample of 15 clusters output by the method. Visual inspection reveals the typical structure of characters decomposed into a left and a right part. This is in no way surprising given the features that we used and the frequency of this structure.

Figure 10 shows two examples of less usual patterns that consist in the addition of some elements in the middle of characters or a longer stroke in the central part of the characters.

Experiments performed with a lesser number of features show examples of clusters where the differentiation between the characters is not sufficient. An example of this is given in Fig. 11. In this example, only the numbers of black pixels on each row

[4] We use a machine with 4 Gb memory equipped with an Intel Core i5 processor at 1.7 GHz.

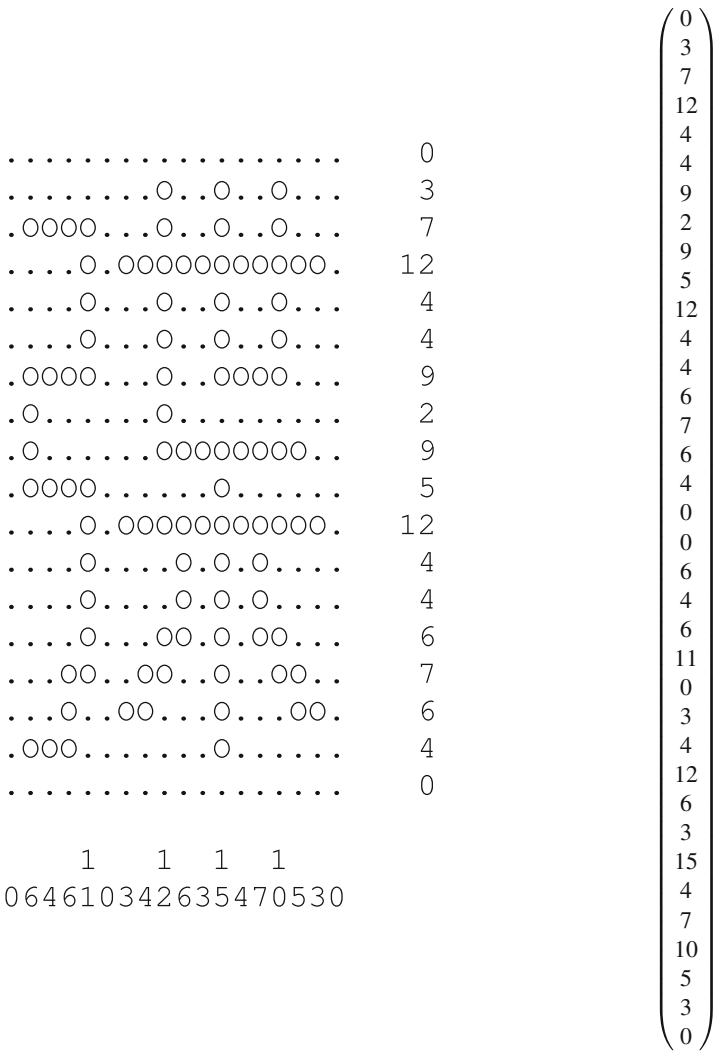

Fig. 8 Values of features for the leftmost character of Fig. 7. The features used here are the number of *black* pixels (represented by a small *circle*) on each row and each column. Eighteen rows and eighteen columns are used, hence a total of 36 values. The values for each row are given to the *right* of the bitmap. The values for each column are given below the bitmap. The vector of feature values for this character is given on the *right*. This vector lists the values for each row followed by the values for each column

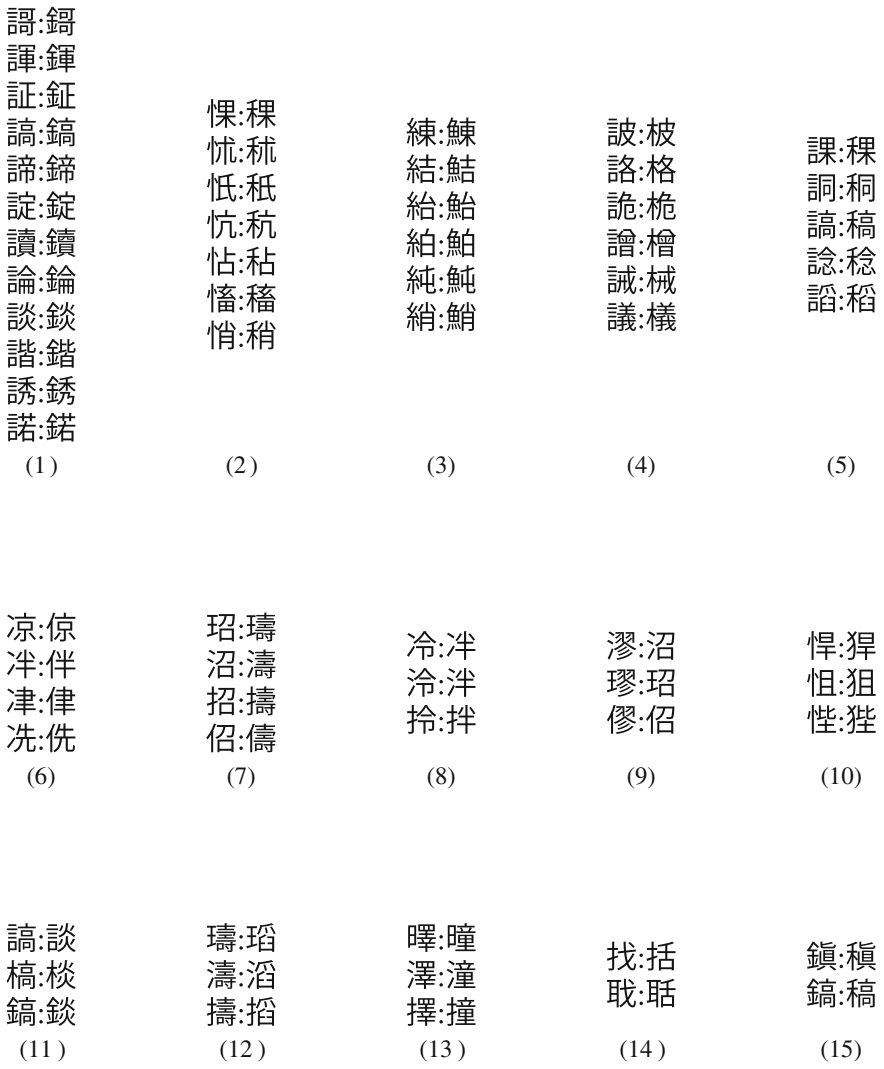

Fig. 9 A sample of 15 analogical clusters output by our clustering method on 14,655 characters from the font knj10B. For Clusters (1) to (6), Cluster (10) and Cluster (15), both characters on the same line share the same *right* part (radical). The *left* parts (keys) are different but common to all lines. Conversely, in the other clusters the *right* part of the characters is the same in each column of the cluster. These clusters show commutations of various keys with only two different radicals

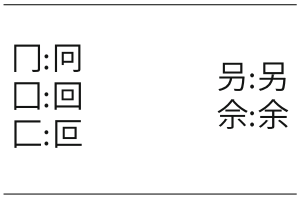

Fig. 10 Two examples of less typical cases of analogical clusters output by our clustering method. The first one on the *left* shows the insertion of a *square* in the *middle* of the character. The second one captures a longer stroke in the center of the characters on the right

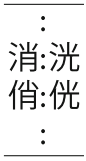

Fig. 11 A cluster output by our clustering method using only the number of *black* pixels on each row as features. Taking into account the columns eliminates this analogy

have been used (18 features). In this case, the vertical directions of the strokes in the left characters on the first and second lines cannot been distinguished so that the method concluded to a proportional analogy that may be questionable or not (for reasons of equivalence between various forms of writing).

The distribution of clusters by number of pairs of objects (36 features) is plotted on Fig. 12. This distribution exhibits a long tail. Few clusters are very large while short clusters are more numerous, which sounds intuitive. The fact that the number of clusters with 3 pairs of characters (451) is greater than the number of clusters with only two pairs of objects (216) is explained by the elimination of redundant clusters reduced to one analogy. The largest cluster contains 55 pairs of characters.

Number of Clusters per Character The number of different characters that appear in at least one analogical cluster was 5,982. This represents 41 % of the total number of characters used (14,655). We *a priori* expected a higher number.

We also measured the number of (non-redundant) clusters each character appears in. Figure 13 plots the distribution of characters per number of clusters they appear in. The use of logarithmic scales suggests a Zipfian distribution that needs more enquiry. This measure gives an estimation of the complexity of a character by the number of oppositions it has with all other characters. This reflects its degree of freedom in the overall graphical system. A character which does not appear in

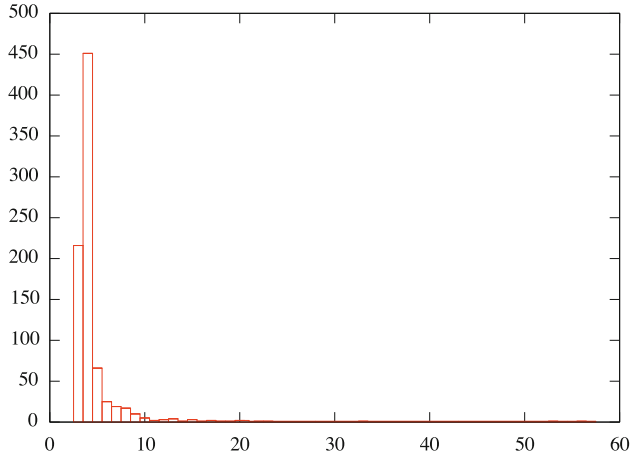

Fig. 12 Distribution of clusters by number of pairs. In abscissae, number of pairs in the clusters. In ordinates, number of clusters with the same number of pairs. The largest cluster contains 55 pairs. There are only 16 clusters between 13 and 55 on the horizontal axis

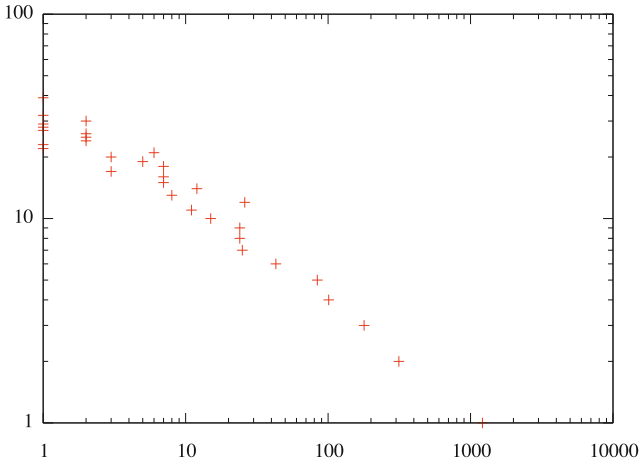

Fig. 13 Distribution of characters by number of clusters they appear in. In abscissae, number of characters that appear in the same number of clusters. In ordinates, number of clusters. Logarithmic scales

any cluster is somehow free relatively to the overall system. From the point of view of acquisition, we hypothesize that characters that appear in more clusters may be easier to remember if the learner has access to the global view given by the clusters. In addition, of course, the number of strokes should be taken into consideration.

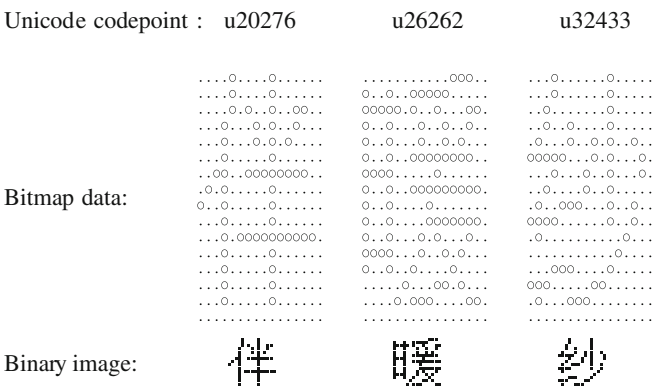

Fig. 14 Visualization of three simplified Chinese characters (伴, 暖 and 纱) from the font FireflyR16 as binary images. In the bitmap data, *white* pixels are visualized as a *dot*, *black* pixels as a small *circle*

6.3 5,000 Most Frequent Simplified Chinese Characters

The font we use in this second series of experiments is FireflyR16.[5] The characters in this font have a fixed height of 16 rows and a fixed width of 16 pixels. Figure 14 visualizes the representation of three randomly selected characters from this font. This font contains traditional and simplified Chinese characters.

From this font, we extracted 4,997 simplified Chinese characters by intersection with a list of the most frequent 5,000 simplified Chinese characters (three characters absent from the font). The frequencies come from the usage on Usenet and the information is available through Unihan data.[6] The 4,997 characters extracted are scattered inside the range of the Unicode codepoints between 19,968 (一) and 40,866 (龥).

Features Used In this second series of experiments, we explore a certain number of features. The set of features is presented in Table 3. We use the same basic features as in the first experiments with the number of black pixels on each row or column (features "lin", "col" and "lincol"). In addition, we tested the use of more elaborated features inspired by classical processing of binary images: number of pixels with a given contextual configuration ("nei") and number of pixels with the same gray level for its neighborhood ("ngr"). A classical data structure on images is that of a quadtree [21]. We use this data structure and count the number of black pixels in each node of the quadtree to form a certain number of sets of features according to the number of levels in the quadtree ("qua"). A feature vector for a specific character and the features "qua" is given in Fig. 15. As intuitively not all pixels are necessary to get an image of the characters represented, we also created a certain number of

[5] See http://wiki.debian.org.hk/w/Make_Debian_support_Chinese.

[6] See http://www.unicode.org/reports/tr38/tr38-10.html.

Table 3 Name and description of the different features used

Name	Number of dimensions of the vectors representing the objects	Possible values
col	16 = number of columns	Number between 0 and 16 of black pixels on each column
lin	16 = number of rows	Number between 0 and 16 of black pixels on each row
lincol	32 = 16 + 16 = number of rows and columns	Number between 0 and 16 of black pixels on each row or column
nei	$512 = 2^9$ = possible number of possible exact configurations around a pixel, this pixel included	Number of pixels, between 0 and $16 \times 16 = 256$, in this configuration
ngr	9 = 8 + 1 = possible number of black pixels around a pixel, the pixel included	Number of pixels, between 0 and $16 \times 16 = 256$, with this number of black pixels around it, itself included
pix	$256/n$ pixel positions selected at random depending on the value of the parameter n:	0 or 1
pix	– 1: 256 = all pixels – 2: 128 = half of the pixels at random – 3: 85 = one third of the pixels – 4: 64 = a quarter of the pixels – 5: 51 = one fifth of the pixels – etc.	
pyx	$256 \times (n-1)/n$ pixel positions selected at random according to the value of the parameter n: – 2: 128 = pix 2 (thus, not used) – 3: 171 = two thirds of the pixels – 4: 192 = three quarters of the pixels – etc.	0 or 1
qua	Number of nodes in the quadtree according to the number of levels in the quadtree: – 0: 1 (too rough, not used here) – 1: 1 + 4 = 5 (too rough, not used here) – 2: 1 + 4 + 16 = 21 – 3: 1 + 4 + 16 + 64 = 85 – 4: 1 + 4 + 16 + 64 + 256 = 341	Number of black pixels in each node of the quadtree, i.e., a number between 0 and 256 for nodes of level 0, between 0 and 64 for nodes of level 1, between 0 and 16 for nodes of level 2, etc.

sets of features by selecting pixel positions at random ("pix" and "pyx"). The actual value of the pixel, black or white, is used as the value of such features.

For each of the features used, we checked whether the space of objects is completely separated by these features, i.e., whether each vector of feature values represents only one object in the set of objects. This is not the case for each set of features and the number of indistinguishable characters, i.e., characters with the same feature vector as some other one, is given in Table 4 for each of set of features. A maximum of 35 indistinguishable characters is observed for "pix 7" which stands for a set of 37 pixel positions selected at random out of 256 possible positions. This shows

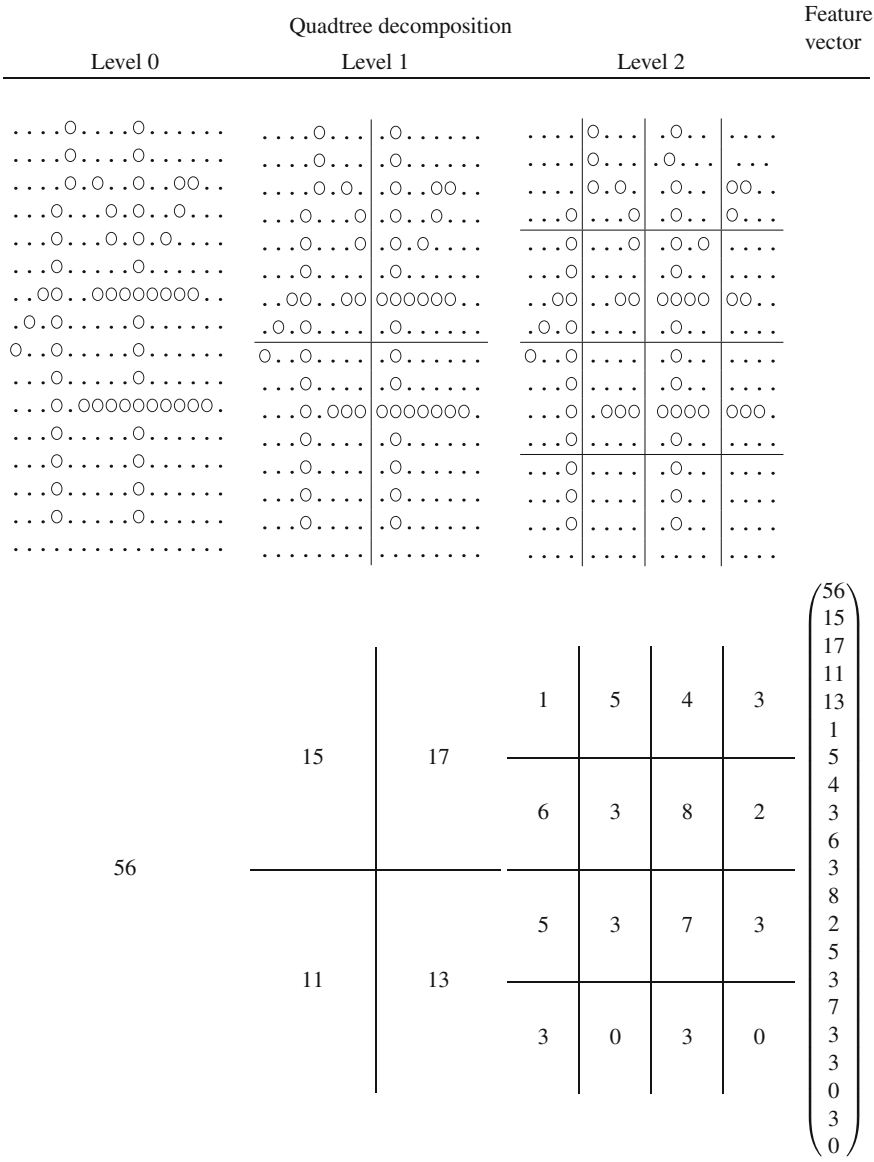

Fig. 15 An example of features used: number of *black* pixels in the quadtree down to level 2 ('qua 2' in Table 3). The leftmost character of Fig. 14 is used. The quads are enumerated one level after another. To build the vector, on each level, the quads are read in the usual reading fashion, i.e., from *left* to *right* and from *top* to *bottom*. The resulting feature vector is given on the *right*

Table 4 Number of characters indistinguishable from another one (doubles) according to each set of features and number of analogies output for each set of features

	col	lin	lincol	nei	ngr	pix 1	pix 2	pix 3	pix 4	pix 5	pix 6	pix 7
# of doubles	1	1	0	1	2	0	2	1	8	16	31	35
# of analogies	14	40	8	0	6042	8	42	43	115	308	4770	5016
	pyx 2	pyx 3	pyx 4	pyx 5	pyx 6	pyx 7	qua 2	qua 3	qua 4			
# of doubles	3	2	0	2	0	0	1	0	0			
# of analogies	40	14	15	10	12	9	22	14	8			

The number of analogies is given here as the analogies (not the clusters themselves) serve as comparison across the experiments with different features (see Table 5)

that the binary images representing these simplified Chinese characters are highly graphically redundant.

Analogical Clusters Obtained We applied out method to extract the analogical graphical structure from the selected 5,000 simplified Chinese characters represented in our selected monospace font. The same program as in the first experiment, run on the same machine used in the first experiment, needed around 5 min for each set of features.

A summary of the number of analogies is given in Table 4. The maximum is obtained for the feature set "pix 7". A visual inspection shows that a lot of analogies obtained do not meet intuition. For an example of such counter intuitive analogy, but obtained with the feature set "pix 5", see Fig. 16.

The best set of clusters according to human judgment is the one obtained with the feature set "pix 5". A sample of the clusters obtained is given in Fig. 16. Again, the typical structure of characters with a left and a right part is revealed.

In order to estimate how similar or different the clusters obtained using the different sets of features are, we performed a systematic comparison. The results of this comparison are given in Table 5. In this table, the sets of clusters obtained by different features are expanded as sets of analogies, i.e., each cluster is expanded into the set of analogies it represents. The results of the experiments are then compared in terms of analogies in common, as the analogies are the final product of the clustering process and as such constitute the final units to compare. The number of clusters would not give such a meaningful comparison. In Table 5, each cell contains three values. The first value gives the percentage of analogies obtained using the set of features given on the row that are included in the set of analogies obtained using the set of features given on the column. The second value gives the same percentage, but with row and column exchanged. The third value is the similarity between the two sets of analogies obtained using the two sets of features, computed as their Jaccard index. For instance, the cell on the line "lincol" and column "col" contains the three values 57 %, 100 % and 0.57. These values should be read as follows: 57 % of the analogies obtained with the "col" features are found using the "lincol" features. All the analogies (100 %) found using the "lincol" features are found using the "col" features. As a result, logically, as this is a case of inclusion of one set of analogies in the other one, the Jaccard index (0.57) is equal to the inclusion percentage 57 % of the smaller set in the larger one.

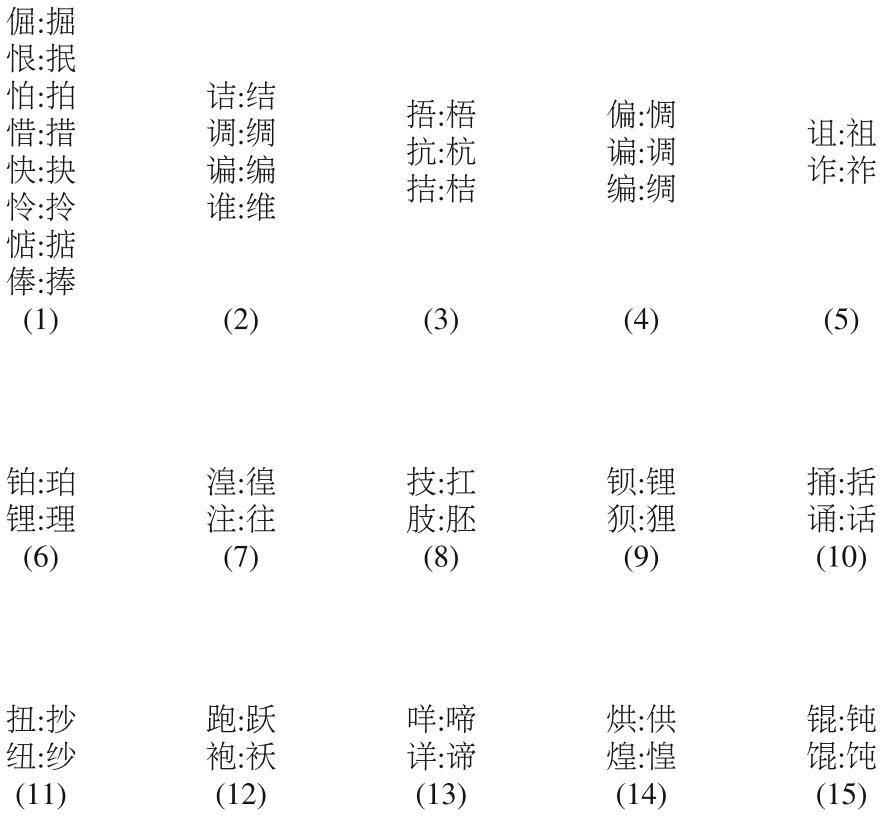

Fig. 16 A sample of 15 analogical clusters output by our clustering method on 5,000 characters from the font FireflyR16 using 51 pixel positions selected at random as features (pix 5 in Table 3). These clusters show commutations of various supposed phonetic keys with two different supposed semantic radicals. Cluster (8) is not acceptable for the human eye. Similarly, the second line in Cluster (1) should have been left out

Different feature configurations were found to deliver the same results. For instance, the features for "pix 1" and "qua 4" had a similarity of 1. This means that both methods find exactly the same analogies. The theoretical explanation for this experimental result is as follows: the comparison in quadtree decomposition of binary images of 16×16 pixels down to four levels ($2^4 = 16$), noted by "quad 4", is equivalent to the comparison at each pixel position, noted by "pix 1". Similarly, although theoretically not equivalent in the general case, "lincol" and "pix 1" were found to be equivalent. The same was found for "pix 2" and "pyx 2". These results are mentioned in the first column of Table 5 by equalities.

A negative but interesting result is that the features using the neighbors of a pixel, either by configuration ("nei") or by level of grey ("ngr") are totally unable to deliver any cluster. We thought that these features would grasp some global view of the characters, by capturing some information about strokes. Indeed a vertical (or

Table 5 Similarity between clusters obtained using different features

	col	lin	nei	ngr	pix 1 = lincol = qua 4	pix 2 = pyx 2	pix 3	pix 4	pix 5	pyx 3	pyx 4	pyx 5	pyx 6	pyx 7	qua 2	qua 3
col																
lin	57 %															
	20 %															
	0.17															
nei	.	.														
	.	.														
	.	.														
ngr	.	.	.													
	.	.	.													
	.	.	.													
pix 1	57 %	20 %	.	.												
= lincol	100 %	100 %	.	.												
= qua 4	0.57	0.20	.	.												
pix 2	57 %	25 %	.	.	100 %											
= pyx 2	19 %	23 %	.	.	19 %											
	0.17	0.14	.	.	0.19											
pix 3	57 %	35 %	.	.	100 %	54 %										
	18 %	32 %	.	.	18 %	53 %										
	0.16	0.20	.	.	0.19	0.37										
pix 4	57 %	30 %	.	.	100 %	38 %	55 %									
	6 %	10 %	.	.	6 %	13 %	20 %									
	0.07	0.08	.	.	0.07	0.11	0.18									
pix 5	64 %	37 %	.	.	100 %	78 %	67 %	24 %								
	2 %	4 %	.	.	2 %	10 %	9 %	9 %								
	0.03	0.05	.	.	0.03	0.10	0.09	0.07								

(continued)

Table 5 continued

	col	lin	nei	ngr	pix 1 = lincol = qua 4	pix 2 = pyx 2	pix 3	pix 4	pix 5	pyx 3	pyx 4	pyx 5	pyx 6	pyx 7	qua 2	qua 3
pyx 3	57 %	20 %	.	.	100 %	28 %	27 %	9 %	4 %							
	57 %	57 %	.	.	57 %	85 %	85 %	78 %	92 %							
	0.40	0.17	.	.	0.57	0.27	0.27	0.09	0.04							
pyx 4	57 %	20 %	.	.	100 %	23 %	20 %	12 %	3 %	64 %						
	53 %	53 %	.	.	53 %	66 %	60 %	93 %	66 %	60 %						
	0.38	0.17	.	.	0.53	0.21	0.18	0.12	0.03	0.45						
pyx 5	57 %	20 %	.	.	100 %	23 %	20 %	7 %	3 %	57 %	53 %					
	80 %	80 %	.	.	80 %	100 %	90 %	90 %	100 %	80 %	80 %					
	0.50	0.19	.	.	0.80	0.24	0.20	0.08	0.03	0.50	0.47					
pyx 6	57 %	25 %	.	.	100 %	23 %	20 %	7 %	3 %	57 %	60 %	80 %				
	66 %	83 %	.	.	66 %	83 %	75 %	75 %	100 %	66 %	75 %	66 %				
	0.44	0.24	.	.	0.67	0.23	0.20	0.08	0.04	0.44	0.50	0.57				
pyx 7	57 %	20 %	.	.	100 %	19 %	18 %	7 %	2 %	57 %	53 %	80 %	66 %			
	88 %	88 %	.	.	88 %	88 %	88 %	100 %	100 %	88 %	88 %	88 %	88 %			
	0.53	0.20	.	.	0.89	0.19	0.18	0.08	0.03	0.53	0.50	0.73	0.62			
qua 2	57 %	42 %	.	.	100 %	19 %	23 %	7 %	4 %	57 %	53 %	80 %	83 %	88 %		
	36 %	77 %	.	.	36 %	36 %	45 %	40 %	59 %	36 %	36 %	36 %	45 %	36 %		
	0.29	0.38	.	.	0.36	0.14	0.18	0.07	0.04	0.29	0.28	0.33	0.42	0.35		
qua 3	57 %	32 %	.	.	100 %	19 %	20 %	6 %	3 %	57 %	53 %	80 %	66 %	88 %	63 %	
	57 %	92 %	.	.	57 %	57 %	64 %	57 %	71 %	57 %	57 %	57 %	57 %	57 %	100 %	
	0.40	0.32	.	.	0.57	0.17	0.19	0.07	0.03	0.40	0.38	0.50	0.44	0.53	0.64	

Null values are replaced with a *dot*. Each cell in the table contains three numbers: the first one is the percentage of analogies obtained using the set of features given on the row that are included in the set of analogies obtained using the set of features given on the column; the second one is the same percentage, but with row and column exchanged; the third one is the similarity between the two sets of analogies obtained using the two sets of features, computed as their Jaccard index. For readability, the cells on the diagonal are left blank but should contain: 100 %, 100 %, 1.0. For compactness, the set of features which were observed identical to another one (common cell containing 100 %, 100 %, 1.0) are not mentioned as independent lines and columns, but by equalities in the first line and the first column

horizontal) stroke includes many black pixels with the same environment of only two black pixels around vertically (or horizontally). Intuitively, such information should be important for Chinese characters. A possible explanation of the failure is that the equality of difference vectors may be too strong a constraint. Relaxing the equality to some approximation could supposedly help, but approximations imply thresholds that are sometimes difficult to justify in all generality.

7 Related Works and Conclusion

This chapter presented a method to automatically extract all possible proportional analogies from a given set of objects represented by feature vectors. The sets of objects we applied this method on were sets of binary images representing Sino-Japanese or Chinese characters in monospace fonts.

The present work is part of a more general research program that attempts at exploring the ways proportional analogy structures language units, following linguistic insights (Varro, Paul, Saussure, Bloomfield, Itkonen). In previous works we have reported on the use of proportional analogy between strings of symbols for various purposes:

- estimation of the number of true analogies, i.e., in form and meaning, between short sentences contained in a corpus from the tourism domain [5];
- estimation of the number of true analogies between chunks in languages of different typologies, Japanese and English [6, 7];
- measure of the proximity of 11 European languages by commonality in the structure of their vocabularies [22];
- use of analogy in machine translation and paraphrasing [23, 24].

The work reported in this chapter is not directly concerned with the linguistic implications of the findings. But it represents one of the extensions of our research program to other types of data structures than strings of symbols, on datasets still in relationship with language. This is in agreement with the first of the two desirable extensions from the data structure of strings of symbols that we have already mentioned in [18].

Strings of symbols being a data structure with only one dimension, a first extension concerns the number of dimensions. The work presented here on images of black and white pixels is an answer to this problem, as binary images are two-dimensional objects on a vocabulary reduced to two symbols: the black pixel and the white pixel. The second extension, not addressed in this chapter, concerns the discrete number of symbols used in strings of symbols. The use of continuous values instead of a discrete number of values is certainly desirable as, for one dimension, strings of continuous values in relation with language are sound waves of spoken utterances.

The object of the first part of the present chapter was to introduce a technique that we have already used in previous works. But we did not give an in-depth description of

Table 6 A cluster of short sentences that illustrates the exchange of predicates: *keep this baggage* versus *draw me a map*. The sentences have been tokenized

Pairs of sentences		
could you keep this baggage ?	:	could you draw me a map ?
keep this baggage, please .	:	draw me a map, please .
will you keep this baggage ?	:	will you draw me a map ?
please keep this baggage .	:	please draw me a map .

Table 7 A cluster of short sentences that illustrates the structural transformation of *i 'd like to ...* into *can i ...here?*

Pairs of sentences		
i 'd like to cash this traveler's check .	:	can i cash this traveler's check here ?
i 'd like to make a hotel reservation .	:	can i make a hotel reservation here ?
i 'd like to make a reservation .	:	can i make a reservation here ?
i 'd like to check my baggage .	:	can i check my baggage here ?
i 'd like to leave my baggage .	:	can i leave my baggage here ?
i 'd like to leave my luggage .	:	can i leave my luggage here ?
i 'd like to reserve a room .	:	can i reserve a room here ?
i 'd like to have dinner .	:	can i have dinner here ?
i 'd like to check in .	:	can i check in here ?
i 'd like to swim .	:	can i swim here ?

The first form is an understatement for a request. The second form is more direct. The sentences have been tokenized

this technique. In these works [5, 19], we applied the method presented in this chapter to short sentences, in complement to the computation of the edit distance constraint which is necessary for proportional analogies of commutation between strings of symbols.[7] This method allowed us to visualize linguistically relevant oppositions between short sentences in English and Japanese, and constructions in line with construction grammars (CxG, [26]). The clusters that we obtained on these data sets illustrated a range of phenomena of different order, like:

- orthographical variations (e.g.: centre : center);
- fronting of interjections (e.g.: Do …, please. : Please do. . .);
- exchange of place names, document names, item names etc.;
- positive vs comparative forms of adjectives (e.g.: green : greener);
- structural transformations, like interrogative vs affirmative;
- exchange of predicates in the same grammatical subject and object context (as illustrated by Table 6);
- questions in different levels of politeness (as illustrated by Table 7);
- etc.

In the present chapter, relying on specific properties of our formalization of objects as feature vectors, we defined the ratio between objects as a difference between vectors, and conformity as equality between difference vectors. This particular setting

[7] Reference [25] is the first mention of the edit distance constraint in terms of similarities; [2] gives the equivalent expression with edit distances; [17] is the published form of the proceedings in which [2] appeared, with few years delay. The edit distance constraint is necessary between strings of symbols to avoid many spurious analogies that would be formed without it.

allowed us to reformulate the problem, which has a complexity of $O(n^4)$, in an equivalent problem with a quadratic complexity, that of enumerating analogical clusters, i.e., lists of pairs of objects with the same ratio (Sect. 4).

We proposed an adequate data structure for this problem (Sect. 3) and, by further exploiting the properties of proportional analogies, we showed how to avoid redundancy in the enumeration and non-informative clusters (Sect. 5). With all this, we showed that the problem becomes tractable so as to solve our problem at hand: extracting all analogies between Sino-Japanese characters in their graphical form, in a reasonable amount of time (Sect. 6).

Although the iconicity of proportional analogies [14, 27] has already been stressed in a broader sense of the word, all the previous approaches to the problem of analogies between images [28–30] are, to our knowledge, based on some high level descriptions of the images, i.e., they rely on a coding of the images. It is usually left unclear how this encoding could be achieved automatically. Thus the previous approaches to the problem are not fully automatized: they do not process the images directly. To our knowledge, this chapter is the first attempt at solving proportional analogies between images directly, under their actual form of binary images of black and white pixels, using their graphical form directly, i.e., without any recourse to high level abstract representations. Our approach, its application to Chinese characters, and our results, show that an explicit description of characters in terms of their constituents; i.e., keys or radicals, as used in [15], can be avoided.

Our proposed method does not exhaust the subject of proportional analogies between binary images made of black and white pixels. There remains a number of problems. One problem is the necessarily fixed size of the icons to compare, hence our use of monospace fixed-size fonts. First, any shift of a character by one or several rows (or columns) would disrupt the analogical relations that are made possible to compute with our feature vectors because almost all characters are well lined up with the first line and column. Second, analogical relations between characters of different sizes cannot obviously be captured with the method proposed here. These two reasons lead to the fact that, for the same fixed-size font, our method is not yet able to identify analogies of the form: 木 : 沐 :: 十 : 汁 . This pleads for the application of scale invariant feature transform techniques, something that we intend to test in the future.

The work presented here is a preprocessing step in a larger study of proportional analogies of graphical form and pronunciation among Chinese characters. We are also performing the same kind of analogical clustering on the level of pronunciation and compute the intersection between analogical clusters on the graphical and on the pronunciation levels. We hypothesized that knowing analogical correspondences between the graphical and pronunciation levels of Chinese or Sino-Japanese characters would ease their memorization by learners. We already performed tests on part of this hypothesis with subjects. We confirmed that the presence of graphical analogies between characters in sets of characters to remember, greatly increases their memorization, when compared with sets of characters containing no analogy [31].

References

1. Gentner, D.: Structure mapping: a theoretical model for analogy. Cogn. Sci. **7**(2), 155–170 (1983)
2. Lepage, Y.: Analogy and formal languages. In: Proceedings of FG/MOL 2001, Helsinki, pp. 1–12 (2001)
3. Yvon, F., Stroppa, N., Miclet, L., Delhay, A.: Solving analogical equations on words. Rapport Technique ENST2004D005, ENST (2004)
4. Hoffman, R.R.: Monster analogies. AI Mag. **11**, 11–35 (1995)
5. Lepage, Y.: Lower and higher estimates of the number of "true analogies" between sentences contained in a large multilingual corpus. In: Proceedings of COLING-2004, vol. 1, pp. 736–742. Geneva (2004)
6. Lepage, Y., Migeot, J., Guillerm, E.: A corpus study on the number of true proportional analogies between chunks in two typologically different languages. In: Proceedings of the Seventh International Symposium on Natural Language Processing (SNLP 2007), pp. 117-122. Kasetsart University, Pattaya, Thailand, ISBN 978-974-623-062-9 (2007)
7. Lepage, Y., Migeot, J., Guillerm, E.: A measure of the number of true analogies between chunks in Japanese. Lect. Notes Artif. Intell. **5603**, 154–164 (2009)
8. Veale, T., Chen, S.: Learning to extract semantic content from the orthographic structure of Chinese words. In: Proceedings of the 17th Irish Conference on Artificial Intelligence and Cognitive Science (AICS2006) (2006)
9. Paul, H.: Prinzipien der Sprachgeschichte. Niemayer, Tübingen (1920)
10. Varro, M.T.: De lingua latina. Coll. Belles-lettres. Trad. J. Collart., Paris (1954)
11. Turney, P.D., Littman, M.L.: Corpus-based learning of analogies and semantic relations. Mach. Learn. **60**(1–3), 251–278 (2005)
12. Turney, P.D.: Similarity of semantic relations. Comput. Linguist. **32**(2), 379–416 (2006)
13. Turney, P.: A uniform approach to analogies, synonyms, antonyms, and associations. In: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pp. 905–912. Coling 2008 Organizing Committee, Manchester, UK (2008)
14. Itkonen, E.: Iconicity, analogy, and universal grammar. J. Pragmat. **22**(1), 37–53 (1994)
15. Yencken, L., Baldwin, T.: Measuring and predicting orthographic associations: modelling the similarity of Japanese kanji. In: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pp. 1041–1048. Coling 2008 Organizing Committee, Manchester, UK (2008)
16. Matsushita, K., Lepage, Y.: Rediscovering the structure of Chinese characters using analogy-based methods (in Japanese). In: Proceedings of the 18th Japanese National Conference in Natural Language Processing, pp. 438–441. Nagoya (2013)
17. Lepage, Y.: Analogy and formal languages. Electron. Notes Theor. Comput. Sci. **53**, 180–191 (2004)
18. Lepage, Y.: Of that kind of analogies capturing linguistic commutations (in French). Habilitation thesis, Joseph Fourier Grenoble University (2003)
19. Lepage, Y., Goh, C.: Towards automatic acquisition of linguistic features. In: Jokinen, K., Bick, E. (eds.) Proceedings of the 17th Nordic Conference on Computational Linguistics (NODALIDA 2009), pp. 118–125. Odense (2009)
20. Langlais, P., Yvon, F.: Scaling up analogical learning. In: Coling 2008: Companion Volume: Posters, pp. 51–54. Coling 2008 Organizing Committee, Manchester, UK (2008)
21. Finkel, R., Bentley, J.: Quad trees: a data structure for retrieval on composite keys. Acta Informatica **4**(1), 1–9 (1974)
22. Lepage, Y., Gosme, J., Lardilleux, A.: Estimating the proximity between languages by their commonality in vocabulary structures. In: Lecture Notes in Artificial Intelligence Human Language Technology—Challenges for Computer Science and Linguistics, pp. 127–138. (2011)
23. Lepage, Y., Denoual, E.: Purest ever example-based machine translation: detailed presentation and assessment. Mach. Transl. **19**, 251–282 (2005)

24. Lepage, Y., Denoual, E.: Automatic generation of paraphrases to be used as translation references in objective evaluation measures of machine translation. In: Proceedings of the Third International Workshop on Paraphrasing (IWP 2005), pp. 57–64. Jeju (2005)
25. Lepage, Y.: Languages of analogical strings. In: Proceedings of COLING-2000, vol. 1, pp. 488–494. Saarbrücken (2000)
26. Croft, W.: Radical Construction Grammar: Syntactic Theory in Typological Perspective. Oxford Linguistics. Oxford University Press, Oxford (2001)
27. Itkonen, E.: Analogy as structure and process: approaches in linguistics, cognitive psychology and philosophy of science. In: Dascal, M., Gibbs, R.W., Nuyts, J. (eds.) Human Cogntive Processing, vol. 14, p. 250. John Benjamins Publishing Company, Amsterdam/Philadelphia (2005)
28. Hofstadter, D.: The Fluid Analogies Research Group: Fluid Concepts and Creative Analogies. Basic Books, New York (1994)
29. Lepage, Y.: Solving analogies on words: an algorithm. In: Proceedings of COLING-ACL'98, vol. I, pp. 728–735. Montréal (1998)
30. Correa, W., Prade, H., Richard, G.: When intelligence is just a matter of copying. In: Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012), pp. 276–281 (2012)
31. Matsushita, K.: Data processing of Chinese Hanzi by proportional analogy and verification of learning efficiency by subjects (in Japanese). Master's thesis, Graduate School of Information, Production and Systems, Waseda University (2013)

http://www.springer.com/978-3-642-54515-3

Computational Approaches to Analogical Reasoning: Current Trends

Prade, H.; Richard, G. (Eds.)
2014, X, 395 p. 105 illus., 18 illus. in color., Hardcover
ISBN: 978-3-642-54515-3