Reusable Lexical Representations for Idioms

Jan Odijk

UIL-OTS, University of Utrecht
Trans 10, 3512 JK Utrecht, The Netherlands
jan.odijk@let.uu.nl

Abstract

In this paper I introduce (1) a technically simple and highly theory-independent way for lexically representing flexible idiomatic expressions, and (2) a procedure to incorporate these lexical representations in a wide variety of NLP systems. The method is based on Structural EQuivalence Classes for Idioms and therefore called the SEQCI method. I illustrate the approach using the Rosetta MT system as an example of an NLP system. I discuss the advantages and some possible objections to the method. I conclude that the method is a good candidate for a standard for the lexical representation of idioms. The method also has the potential to be used for multi-word expressions other than idioms.

1. Introduction

State-of-the art NLP systems do not deal adequately with large numbers of multi-word expressions (MWEs), and this forms a major obstacle for the successful application of NLP technologies in domains such as information retrieval and summarization, question answering and machine translation.[1] MWEs pose many problems, among them their automatic identification in text (without the use of a grammar), their treatment in a grammar, and their lexical representation.

This paper focuses on the lexical representation of MWEs. Since MWEs form a heterogeneous class, I restrict attention in this paper to *flexible idioms*. Flexible idioms are called *flexible* because the order of their components is not fixed and other words or phrases can intervene between their components. Dutch *de plaat poetsen* (lit. polish the plate, 'to bolt') can be used to illustrate this (see (1) in which the bold parts are part of the idiom):

(1) a. Hij heeft gisteren **de plaat gepoetst**
lit. 'He has yesterday the plate polished'
b. Ik dacht dat hij gisteren **de plaat** wilde **poetsen**
lit. 'I thought that he yesterday the plate wanted polish'
c. Hij **poetste de plaat**
lit. 'He polished the plate'
d Hij **poetste** gisteren **de plaat**
lit. 'He polished yesterday the plate'

By assigning a flexible idiom the syntactic structure that it would have as a literal expression, it will participate in the syntax as a normal expression, and permutations, intrusions by other words or phrases, etc. can occur just as they can occur with these words in their literal interpretation.

Flexible idioms often have restrictions on their syntactic behavior additional to the ones on non-idiomatic constructions. Many of these restrictions can be predicted from general principles (given an adequate description of the idioms) and should therefore follow from the design of the grammar used in an NLP system (see (Schenk, 1994) for one approach). Other restrictions on idioms cannot always be reduced to general grammatical properties or principles, and must be stipulated as idiosyncratic properties, e.g. passivization.

2. SEQCI

Though a flexible idiom requires a syntactic structure in its lexical specification, the core idea behind the SEQCI method is that it does not describe what structure an idiom has, but backs off to a slightly weaker position and describes which idioms have the same structure, thus creating structural equivalence classes for idioms.

The SEQCI method consists of two parts: (1) it describes a specific way of representing the formal aspects of idioms in a lexicon. This lexical representation is technically very simple and highly theory-independent; an essential characteristic is that it partitions idioms into equivalence classes. (2) The method provides a procedure that allows one to incorporate all members of an equivalence class into one's NLP system in a fully automatic manner after having manually incorporated one instance of this equivalence class.

Lexical representation. (1) Each idiom is assigned to a structural equivalence class, by assigning it an idiom pattern. Additionally, (2) a list of the citation forms of all the words making up the idiom (the 'idiom component list', ICL) is specified. The order of the words on this list is free, but it has to be identical for all members of the same equivalence class. Finally, (3) for each idiom an example sentence has to be provided, with the condition that the syntactic structures of all example sentences from a single equivalence class are identical (abstracting from specific lexical items). Each equivalence class is described by free text comments, describing and justifying the structure, differentiating it from other equivalence classes, etc. This free text format will only be used by humans.

Procedure for incorporation in to an NLP system. Given a set of idioms described in this manner, one can incorporate them into one's own NLP system by the following procedure, which consists of a manual and an automatic part.

Manual part. The manual part of the conversion procedure for a given equivalence class EQ1 consists of 5 steps: (1) Have the example sentence of an arbitrary instance of EQ1

[1](Sag et al., 2001), (Thurmair, 2003), (Dowdall et al., 2003), (Odijk, 2000), (Akkermans et al., 2004), p.118

parsed by the system, resulting in a reference parse; (2) Define a transformation ('parse transformation', PT) to turn the reference parse into the idiom structure; (3) determine the idiom component identifier list (ICIL), i.e. list of unique identifiers of the lexical items used in the idiom, from the reference parse fringe; (4) Use the ICIL to define a transformation ('idiom component list transformation', ICLT) to remove and/or reorder lexical items in the ICL; (5) Apply the ICLT to the ICL resulting in a 'transformed idiom component list' (TICL) and check that the citation form of each element of the ICIL equals the corresponding element on the TICL.

Automatic part. The automatic part of the conversion procedure is applied to each instance of equivalence class EQ1, and also consists of 5 steps: (1) Select an instance from EQ1, have its example sentence parsed by the NLP system, and check that the resulting structure is identical to the reference parse, except for the lexical items; (2) Use the PT to turn this parse tree into the structure of the idiom; (3) Determine the ICIL; (4) Apply the ICLT to the ICL, yielding the TICL; (5) Check that the citation form of each element of the ICIL equals the corresponding element on the TICL.

3. Illustration

We illustrate the SEQCI method by showing how given a lexical description of idioms in accordance with the SEQCI method we can derive the correct representations required in the Rosetta machine translation system (Rosetta, 1994). Suppose we have an idiom pattern description for an idiomatic expression such as *het ijs breken* 'to break the ice':

(1) **idiom pattern** MWEp1

Comments expression headed by a verb taking a direct object NP that consists of a determiner and a singular noun, and taking one argument realized as the subject.

And suppose that we have instances of idiom pattern MWEp1:

(2)

Idiom components	Example sentence
het ijs breken	Hij heeft het ijs gebroken
de toon aangeven	Hij heeft de toon aangegeven
de plaat poetsen	Hij heeft de plaat gepoetst
...	...

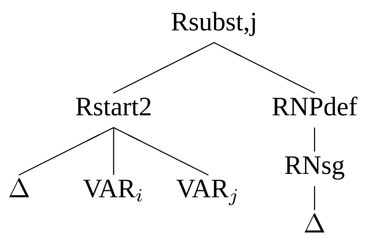
etc. for all examples in (3):

(3) *de toon aangeven, het woord voeren, het woord vragen, het leven laten, de kar trekken, de boot afhouden, de boot missen, het gelag betalen, de plaat poetsen, de scepter zwaaien, het onderspit delven, de mammon dienen, de dans ontspringen, de wapenrok dragen, de aftocht blazen, de handschoen opnemen*

The Rosetta system requires the following ingredients in order to adequately deal with the idiom *het ijs breken*: (1) The syntactic structure (4):[2]

[2]This is a simplified representation. The Δ symbols represent slots for the lexical components of the expression

(4)



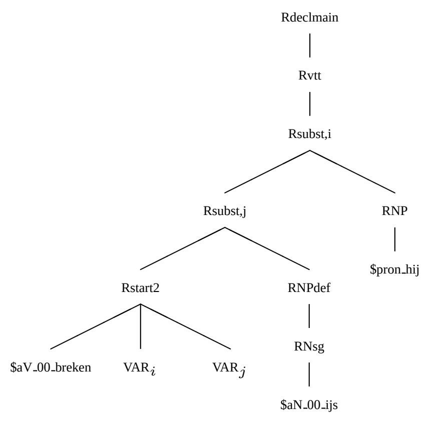
(2) The lexical components making up this expression are the relevant lexical entries for the verb *breken* and the word *ijs* in the Rosetta lexicon. These can be uniquely identified by the keys *$aV_00_breken* (to be distinguished from *$aV_01_breken*, which represents the intransitive variant) and *$aN_00_ijs*. Note that the article *het* is not a lexical component: it is introduced syncategorematically by the rule *RNPdef* that creates definite NPs in the grammar of the Rosetta system.

Finally, (3), it must be specified that a copy of the lexical entry uniquely identified by *$aV_00_breken* must be substituted for the left-most Δ, and a copy of *$aN_00_ijs* for the right-most one. This is done by listing the keys in a specific order: *$aV_00_breken $aN_00_ijs*. This list is this idiom's ICIL.

We start with the manual part of the incorporation procedure.

Manual part. Applying this procedure to the example sentence of the first instance of MWEp1 ('Hij heeft het ijs gebroken') in the Rosetta system yields the parse tree (5):

(5)



The PT to obtain the idiom structure (4) is simple: delete everything above the rule *Rsubst,j* and replace the lexical items by Δ. Determining this transformation requires human intervention. Note that by having the system parse the sentence, unique references to the system's lexical entries for words have been obtained.

The ICL *het ijs breken* must be converted to a list of the citation forms of the keys *$aV_00_breken* and *$aN_00_ijs* (in this order), i.e. *breken ijs*, which can be achieved by the ICLT *1 2 3 $\Rightarrow$ 3 2*, i.e delete the first element and reverse the remaining two elements.

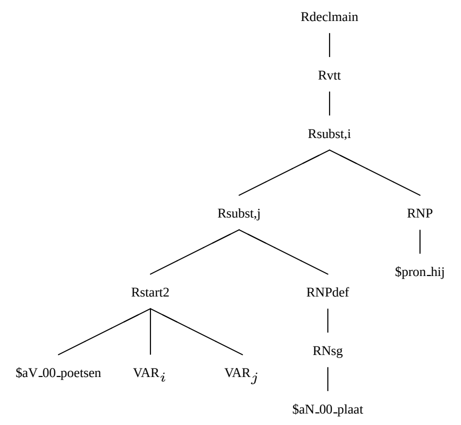
The citation forms of *$aV_00_breken* and *$aN_00_ijs* are indeed *breken* and *ijs*, as required. This concludes the manual part. We now turn to the automatic part.

Automatic part. The automatic part of the conversion procedure is applied to each instance of the equivalence class.

We illustrate the automatic part by applying it to the idiom *de plaat poetsen*.

Parsing the example sentence *Hij heeft de plaat gepoetst* yields the parse tree (6), which is indeed identical to the reference parse (5) modulo the lexical items:

(6)



PT is applied to this tree and indeed turns this tree into idiom structure (4).

Applying the ICLT to *de plaat poetsen* yields the TICL *poetsen plaat*. The ICIL of (6) equals *$aV_00_poetsen $aN_00_plaat*. Computing the citation form for each of the identifiers on this list yields the list *poetsen plaat*, which is identical to TICL. In this way we have obtained all information required by the Rosetta system to incorporate the idiom *de plaat poetsen* in a fully automatic manner. Applying the same procedure to the other idioms of this equivalence class will incorporate all these idioms into the Rosetta system in a fully automatic manner.

4. Potential Objections

In the preceding section we illustrated the procedure to derive a system-specific representation for flexible idioms from the proposed lexical representation. But we illustrated an idealized case only. There are many steps in the procedure that could yield other results than the ones illustrated, both in the manual and in the automatic part. In this section we will discuss these.

The first step is to select an example sentence illustrating the idiom pattern, and to parse it. The resulting parse tree is then used in the further steps. If there is exactly one parse tree, it has to be checked whether this is the system-specific parse from which the idiom structure can be derived. If it is, there is no problem, but what if it isn't? Or what if the example sentence does not yield a parse at all? This may certainly happen, and if it happens, one has to investigate the cause. These can be manyfold, but basically it means that the coverage of the lexicon and/or grammar is insufficient. In fact, it is actually a virtue of the SEQCI method that one is pointed out that the system cannot handle this idiom: it makes no sense to add it if it cannot lead to a correct parse anyway. The remedy is simple: extend the system's lexicon and/or grammar so that it does yield a correct parse.

One case requires special mentioning: though idioms generally have regular syntactic structures, many idioms use structures only allowed in idioms but not in non-idiomatic constructions. Examples are the use of singular count nouns in determinerless NPs (e.g. English *he was afraid of losing face*),[3] the use of inalienable possession constructions in Dutch, examples such as Dutch ***ten** tijde van* 'at the time of' (containing the fossilized portmanteau word *ten* and the *e*-form of the noun *tijd*, both only used in idioms), etc. etc. If such structures recur in many idioms they can be dealt with by having *minor rules* in one's system to describe such structures: minor rules are rules that only can be used to form idiomatic structures. The system should be used in a mode that allows minor rules to be applied for non-idiomatic structures as well when applying the SEQCI method.

It is also possible (actually, probably the most frequent case) that the parser yields multiple structures. Again, it is possible that none of these multiple structures is the one corresponding to the idiom structure. In this case, one will have to extend the lexicon and/or grammar again. Ignoring this case further, there are two possibilities: (1) the structure is ambiguous in aspects not directly related to the idiom, but only to the parts of the example sentence added. In that case an arbitrary selection can be made; (2) the ambiguity concerns the idiom part of the sentence. E.g., a sequence d-n-p-n can be parsed as NP or as NP PP. In this situation the free text comment describing the idiom pattern, and contrasting it with other idiom patterns should help the developer select the right parse.

It may also be the case that the developer, giving his/her knowledge of the system and the idiom pattern description, concludes that the current pattern collapses idioms in a single equivalence class while his/her own system requires a further subdivision. Though this complicates matters, it cannot be an argument against the method proposed here. What we see here is that the proposed method is not completely theory-neutral. However, the same problem would also arise in any other proposal, and in particular if such a proposal describes idiom structures it will arise much more often. The current proposal, however, is –in my view – the most theory-neutral possible.

5. Extensions and Improvements

Several extensions and improvements of the SEQCI method are possible. In this section I briefly mention one. It extends the SEQCI method with parameters. Lack of space prevents me from fully elaborating, formalizing and illustrating this proposal or discuss others.

The extension introduces parameters in the SEQCI method and contributes to reducing the number of equivalence classes and increasing the number of members within equivalence classes. It will therefore reduce the number of idioms that have to be dealt with manually and increase the number of idioms that can be incorporated into an NLP system in a fully automatic manner.

A concrete example may help illustrate this. Idioms can contain nouns. In Dutch, nouns can be singular (sg) or plural (pl) , and positive (pos) or diminutive (dim). In the original SEQCI proposal a different equivalence class would be needed for each of these 4 cases (and even more if more than one noun occurs in a single idiom). By introducing

[3]This example was suggested by an anonymous reviewer.

2 parameters for nouns (sg/pl, pos/dim), it is possible to group these 4 equivalence classes into a single equivalence superclass, and to have a single transformation for this superclass, which however is parameterized for the properties of the noun (sg/pl; pos/dim).

The extension with parameters introduces a little more theory and implementation specificity to the SEQCI method, but it does so in a safe way: NLP systems that can make use of these parameters will profit from it, while systems that cannot make use of these parameters are not harmed since the original equivalence classes can still be identified. For the example given above the theory /implementation dependency that is introduced is that properties such as sg/pl and pos/dim on a noun are dealt with by rules applying to just the noun. It can be expected that many different grammatical frameworks share this assumption.

6. Discussion

For many NLP systems, the proper treatment of flexible idioms requires (1) specification of the syntactic structure of the idiom in this system; (2) unique references to the lexical items for the words occurring in the idiom in the lexicon of the system; (3) a proper link between the lexical items and the syntactic structure. All these aspects are highly system-specific (e.g., as in the Rosetta example), so a lexical representation for a flexible idiom for one system cannot be easily (or at all) be used for other systems.

The SEQCI method overcomes this problem by providing a lexical representation of flexible idioms that abstracts from system-specific, theory-specific and grammatical-framework-specific aspects, and at the same time provides a method to compute these aspects for a given NLP system.

The method proposed here categorizes flexible idioms into equivalence classes. The successfulness of this method will depend on (1) how many different equivalence classes must be distinguished (the less the better), and (2) how many instances each equivalence class contains (the more the better).

We carried out measurements on two databases of idioms to determine this. The first database is a small database of 893 Dutch idioms categorized into parameterized equivalence classes. The second database is the SAID database (Kuiper et al., 2003), in which we approximate such a classification by assuming that the delexicalized syntactic structures of this database correspond to parameterized equivalence classes. The table at the end of the text presents the major findings of our measurements. The result, though not definitive, is promising. It means, e.g., that 80% (or 11,773) of the idioms in the SAID database can be dealt with by just 481 equivalence classes.

7. Conclusions

In this paper I introduced the SEQCI method, which allows one to lexically represent flexible idioms in a highly theory- and implementation-independent way and to incorporate them efficiently in a wide variety of NLP systems. This makes it a suitable candidate for a standard for the lexical representation of idioms. The incorporation procedure was illustrated using the Rosetta MT system. The method makes crucial use of idiom equivalence classes. Initial measurements on the number and size of these equivalence classes are promising.

Though the initial results are promising, it is clear that establishing the feasibility of the method and the reusability of the representations requires further experiments, with a large number of idioms and a wide variety of NLP systems. Though the method has been illustrated for flexible idioms only, I expect that the method will also be useful for other types of MWEs, and might work even better for these. Investigating these issues is a matter for future research.

	SAID		**Dutch**	
Cov.	**#idioms**	**#patterns**	**#idioms**	**#patterns**
50%	7383	28	449	21
60%	8853	54	539	36
70%	10304	140	628	59
80%	11773	481	716	98
85%	12509	908	760	134
90%	13245	1644	804	178
95%	13981	2380	849	223
100%	14716	3116	893	267

8. References

Akkermans, Jaap, Brigit van Berkel, Chris Frowein, Linda van Groos, and Dirk Van Compernolle, 2004. Technologieverkenning Nederlandse taal- en spraaktechnologie. Technical report, M&I/Partners, Amersfoort/Leuven. Rapport bij project 103185, versie 01.

Dowdall, James, Fabio Rinaldi, Fidelia Ibekwe-SanJuan, and Eric SanJuan, 2003. Complex structuring of term variants for question answering. In Francis Bond, Anna Korhonen, Diana McCarthy, and Aline Villavicencio (eds.), *ACL-03 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment. Proceedings of the Workshop*. http://www.aclweb.org: Association for Computational Linguistics.

Kuiper, Koenraad, Heather McCann, Heidi Quinn, Therese Aitchison, and Kees van der Veer, 2003. SAID: A syntactically annotated idiom dataset. Linguistic Data Consortium, LDC2003T10, Pennsylvania.

Odijk, Jan, 2000. Multilingual lexicons. ISLE Meeting, Paris.

Rosetta, M.T., 1994. *Compositional Translation*, volume 273 of *Kluwer International Series in Engineering and Computer Science (Natural Language Processing and Machine Translation)*. Dordrecht: Kluwer Academic Publishers.

Sag, Ivan A., Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger, 2001. Multiword expressions: A pain in the neck for NLP. *LinGO Working Paper*, (2001-03). Http://lingo.stanford.edu/csli/pubs/WP-2001-03.ps.gz.

Schenk, André, 1994. *Idioms and Collocations in Compositional Grammars*. Ph.D. thesis, University of Utrecht.

Thurmair, Gregor, 2003. Industrial requirements for cross-language retrieval. Enabler Workshop, Paris, http://www.enabler-network.org/final-workshop.htm.