You shall find the target via its companion words: specifications of a navigational tool to help authors to overcome the tip-of-the-tongue problem

Michael Zock\textsuperscript{1} Dan Cristea\textsuperscript{2,3}

\begin{itemize}
\item \textsuperscript{1} Aix-Marseille Université, CNRS, LIF UMR 7279, 13000, Marseille, France
  \texttt{michael.zock@lif.univ-mrs.fr}
\item \textsuperscript{2} “Alexandru Ioan Cuza” University of Iaşi
\item \textsuperscript{3} Institute of Computer Science, Romanian Academy
  \texttt{dcristea@info.uaic.ro}
\end{itemize}

Abstract: The ability to retrieve ‘words’ is a prerequisite for speaking or writing. While this seems trivial as we succeed most of the time, it is not simple at all and it can become quite annoying if ever we fail or require too much time. The resulting silence puts pressure on the speaker’s and the listener’s mind, as it disrupts the fluency of encoding (planning what to say) and decoding (interpretation of the linguistic form). When lacking information for a given ‘word’ (for example, its form), we tend to reach for a dictionary. While this works generally quite well for the language receiver, this is not always the case for the language producer. This may be due to a number of reasons like input presentation (underspecification), organisation of the lexicon, etc. We present here the roadmap of a lexical resource whose task it to help authors to find the word they are looking for. More precisely, we present here a framework for building a tool to support word access. To reach our goal several problems need be solved: ‘search space reduction’, ‘clustering the words retrieved in response to some input (available information)’ and ‘labeling the clusters’ to ease navigation. Before starting to build this navigational tool, we define a set of criteria that need to be satisfied by the resources to be used. Next we discuss some of them to see whether they comply with respect to our goal. While being preliminary work, this is clearly a necessary step for building the tool we have in mind.

1. The problem: how to find the word that is eluding you?

One of the most vexing problems in speaking or writing is that one knows a given word, yet one fails to access it when needed. Suppose, you were looking for a word expressing the following ideas: ‘superior dark coffee made of beans from Arabia’, but could not retrieve the intended form ‘mocha’. What will you do in a case like this? You know the meaning, you know how and when to use the word, and you even know its form, since you’ve used it some time ago, yet you simply cannot access it at the very moment of speaking or writing? Since dictionaries generally contain the target word, they are probably our best ally to help us find the form we are looking for. This being said, storage does not guarantee access. The very fact that a dictionary
contains a word does not guarantee at all that we will also be able to find or locate it (Zock & Schwab, 2013; Tulving & Pearlstone, 1966).

Dictionary users typically pursue one of two goals (Humble, 2001): as decoders (reading, listening), they are generally interested in the meanings of a specific word, while as encoders (speakers, writer) they wish to find the form expressing an idea or a concept. This latter task is our goal. While most dictionaries satisfy the reader's needs, they do not always live up to the authors' expectations, helping them to find the elusive word.¹ There are various reasons for this. Some of them are related to the input:

(a) Input specification: What information should one provide to look up a specific word? This is not a trivial issue, even for quite common words, say, 'car', 'apple' or 'elephant. Should the input be a single word, a set of words ('huge, gray, Africa', in the case of elephant), a more general term (category, animal), or a textual fragment (context) from which only the target is missing? Concerning this last point, see for example the shared task of SemEval, devoted to lexical substitution (McCarthy & Navigli, 2009).

(b) Synonymy or term-equivalence: suppose your target (chopstick) were defined as 'instrument used for eating', yet you used the query term 'tool' instead of 'instrument'.

(c) Ambiguity: how does the machine (lexical resource) 'know' which of the various senses you have in mind ('mouse/device' vs. 'mouse/rodent')?²

Others are related to the output produced in response to some input:

(a) Number of outputs: since entries (query terms) can be very broad, i.e. linguistically underspecified (suppose you were to use 'animal' in the hope to find 'elephant'), the number of hits or outputs can be huge. Hence we must organize them. But size can become critical even if one uses other search strategies, as we will do. We try to find the target via its associated term. Since in both cases the list of outputs is huge, we must address this problem, and we believe that the answer lies in clustering or organization.

¹ To be fair, one must admit though that great efforts have been made to improve the situation. In fact, there are quite a few onomasiological dictionaries. For example, Roget’s Thesaurus (Roget, 1852), analogical dictionaries (Boissière, 1862, Robert et al., 1993), Longman’s Language Activator (Summers, 1993) various network-based dictionaries: WordNet (Fellbaum, 1998; Miller et al., 1990), MindNet (Richardson et al., 1998), and Pathfinder (Schvaneveldt, 1989). There are also various collocation dictionaries (BBI, OECD), reverse dictionaries (Kahn, 1989; Edmonds, 1999) and OneLook which combines a dictionary, WordNet, and an encyclopedia, Wikipedia (http://onelook.com/reverse-dictionary.shtml). A lot of progress has been made over the last few years, yet more can be done especially with respect to indexing (the organization of the data) and navigation. Given the possibilities modern computers offer with respect to storage and access, computational lexicography should probably jettison the distinctions between lexicon, encyclopedia, and thesaurus and unify them into a single resource.

² Note that outputs can also be polysemous, but ambiguity is not really a problem here, as all the dictionary user wants is to find a given word form.
(b) *Cluster names*: outputs must not only be grouped, but the groups need names, as otherwise the user does not know in what direction to go, that is, in what bag to look for the target.

Search failure (failing to find) is called *dysnomia* or *Tip of the Tongue-problem* (Brown & McNeill, 1996) if the searched objects are words. Yet, this kind of problem occurs not only in communication, but also in other activities of everyday life. Being basically a search problem, it is likely to occur whenever we look for something that exists in real world (objects) or our mind: dates, phone numbers, past events, peoples' names, or 'you-just-name-it'.

As one can see, we are concerned here with the problem of words, or rather, how to find them in the place where they are stored: lexical resource (dictionary or brain). We will present here some ideas of how to develop a tool in order to help authors (speaker/writer) to find the word they are looking for. While there are various search scenarios, we will restrict ourselves here only to cases where the searched terms exists in the base. Our approach is based on psychological findings concerning the mental lexicon (Levelt, 1989). Hence we draw on notions such as association (Deese, 1965), associative network (Schvaneveldt, 1989) or neighborhood (Vitevitch, 2008). We also take into account notions such as storage (representation and organization), access of information (Roelofs, 1992; Levelt et al., 1999), observed search strategies (Thumb, 2004) and typical navigational behavior (Atkins, 1998). Our goal is to develop a method allowing people to access words, no matter how incomplete their conceptual input may be. To this end we try to build an index, i.e. a semantic map allowing users to find a word via navigation.

2. Search strategies function of variable cognitive states

Search is always based on knowledge. Depending on the knowledge available at the onset one will perform a specific kind of search. Put differently, there are different information needs as there are different search strategies.

There are at least three things that authors typically know when looking for a specific word: its meaning (definition) or at least part of it (this is the most frequent situation), its lexical relations (hyponymy, synonymy, antonymy, etc.), and the collocational or encyclopedic relations it entertains with other words (Paris-city,

---

3 The tip-of-the-tongue phenomenon ([http://en.wikipedia.org/wiki/Tip_of_the_tongue](http://en.wikipedia.org/wiki/Tip_of_the_tongue)) is characterized by the fact that the author (speaker/writer) has only partial access to the word s/he is looking for. The typically lacking parts are phonological (syllables, phonemes). Since all information except this last one seems to be available, and since this is the one preceding articulation, we say: the word is stuck on the *tip of the tongue* (TOT, or TOT-problem).

4 While paper dictionaries store word forms (lemma) and meanings next to each other, this type of information is distributed across various layers in the mental lexicon. This may lead to certain word access problems. Information distribution is supported by many empirical findings like *speech errors* (Fromkin, 1980), studies in *aphasia* (Dell et al., 1997), experiments on *priming* (Meyer & Schvaneveldt, 1971) or the *tip of the tongue phenomenon* (Brown & McNeill, 1996). For computer simulations see (Levelt et al., 1999; Dell, 1986).
Paris-French capital, etc.). Hence there are several ways to access a word (see Figure 1): via its meaning (concepts, meaning fragments), via syntagmatic links (thesaurus-or encyclopedic relations), via its form (rhymes), via lexical relations, via syntactic patterns (search in a corpus), and, of course, via another language (translation).

Suppose you were looking for a (word)form expressing the following ideas 'spring, typically found in Iceland, discharging intermittently hot water and steam'. The corresponding word, 'geyser' or 'geysir' can be recovered in various ways:

1. directly based on its meaning (this is the golden, i.e. normal route). Note that google does quite well for this kind of input. We will also be able to do that, but we can do a lot more, namely, reveal the target via its associates.
2. by searching in the corresponding semantic field: 'hotsprings' or 'natural fountains encountered in Iceland', etc;
3. by relying on encyclopaedic information (co-occurrences, associations):
   - hotspring, Iceland, eruption, Yellowstone;
4. by considering similar sounding words (Kaiser ⇨ geyser);
5. via a lexical relation (synonym, hypernym) like «fountain», hotspring;
6. via a syntactic pattern (co-occurrence): 'spring typically found in Iceland,'
7. translation equivalent (間歇噴泉 ⇨ geyser);

We will consider here only one strategy, the use associations (mostly, encyclopaedic relations). Note that, people being in the TOT-state clearly know more than that. Psychologists who have studied this phenomenon (Brown & McNeill, 1966; Brown, 2012 ; Díaz et al. 2014 ; Schwartz and Metcalfe, 2011) have found that their subjects had access not only to meanings (the word’s definition), but also to information concerning grammar (for example, gender, see Vigliocco et al. 1997) and lexical form: sound, morphology (part of speech). While all this information could be

---

* This feature of the mental lexicon (ML) is very important, as in case of failure of one method, one can always resort to another.
used to constrain the search space, —the ideal dictionary being multiply indexed,— we will deal here only with semantically related words (associations, collocations in the large sense of the word). Before discussing how such a dictionary could be built and used, let us consider a possible search scenario.

We start from the assumption that in our mind, all words are connected, the mental lexicon (brain) being a network. This being so, anything can be reached from anywhere. The user enters the graph by providing whatever comes to his mind (source-word), following the links until he has reached the target. As has been shown (Motter et al. 2002), our mental lexicon has small-world properties: very few steps are needed to get from the source-word to the target word. Another assumption we make is the following: when looking for a word, people tend to start from a close neighbour, which implies that users have some meta-knowledge containing the topology of the network (or the structure of their mental lexicon): what are the nodes, how are they linked to their neighbours, and what are more or less direct neighbours? For example, we know that ‘black’ is related to ‘white’, and that both words are fairly close, at least a lot closer than, say, ‘black’ and ‘flower’.

Search can be viewed as a dialogue. The user provides as input the words that a concept he wishes to express evokes, and the system displays then all (directly) connected words. If this list contains the target search stops, otherwise it will continue. The user chooses a word of the list, or keys in an entirely different word. The first part described is the simplest case: the target is a direct neighbour. The second addresses the problem of indirect associations, the distance being bigger than 1.

3. Architecture and roadmap

As mentioned already, when experiencing word access problems we expect help from dictionaries, hoping to find the elusive term there. Unfortunately, so far there is still not yet a satisfying resource allowing authors (people being in the ‘production mode’: speakers/writers) to find easily and most of the time the resisting word. While WordNet or Roget’s Thesaurus are helpful in some cases, more often than one might think, they are not. This is a problem we would like to overcome. Figure 2 displays in a nutshell our approach, word access being viewed (basically) as a two-step process: two for the user, and two for the resource builder. The task is basically finding a specific item (target word) within the lexicon. Put differently, the task is to reduce the entire set (all words contained in the lexicon) to one, the target. Since it is out of question to search in the entire lexicon, we suggest to reduce the search space in several steps, basically two.
Hypothetical lexicon containing 60,000 words

Given some input the system displays all directly associated words, i.e., direct neighbors (graph), ordered by some criterion or not.

Tree designed for navigational purposes (reduction of search-space). The leaves contain potential target words and the nodes the names of their categories, allowing the user to look only under the relevant part of the tree. Since words are grouped in named clusters, the user does not have to go through the whole list of words anymore. Rather he navigates in a tree (top-to-bottom, left to right), choosing first the category and then its members, to check whether any of them corresponds to the desired target word.

Fig. 2: Lexical access as a two-step process
The goal of the first step is to reduce the initial space (about 60,000 in the case of EAT) to a substantially smaller set. EAT\(^6\) is an association thesaurus which generates all directly related words to some input (the word given by the user, word coming to his mind when trying to find the form of a given concept).

The goal of the second step is to reduce further the search space. Since the list of words directly associated to the input is still quite huge (easily 150-500 words), we suggest to cluster and label the terms to build a categorial tree. This allows the user to search only in the relevant part of the categorial tree, rather than in the entire list, leaving him finally with a fairly small number of words. If all goes well, he will find the target in this tree, otherwise, he will have to iterate, either starting from an entire new word or choosing one contained in the selected cluster.

Note, that in order to display the right search space, i.e. set of words within which search takes place (step-1), we must have already well understood the input — [mouse (rodent) vs. mouse(device)] — as otherwise our set may contain many (if not mostly) inadequate candidates: ‘cat/cheese’ instead of ‘computer/screen’ or vice versa. Note also, that the ideal resource should allow us to solve both problems: allow for navigation in an associative network while presenting the potential candidates in meaningfully named clusters (categorial tree). Given the complexity of the task at hand, we will certainly not try to start building such a resource. Rather we will try to ‘discover’ the best one among the existing resources. Hence, we will propose here below a methodology to evaluate the advantages and shortcomings of a number of notorious resources in correlation with our problem at hand.

4. Formalisation of resources and discussion

To evaluate the adequacy of the various resources to be considered when building our tool requires some criteria, which we call properties. Let’s note that some of them are binary (yes/no) hence likely to act as constraints, while others are gradual, they could be valued. Here is an initial set of such properties:

1. **Representation**: undirected graph (constraint). In this graph vertexes are words\(^7\) (possibly ambiguous), or word senses\(^8\) (therefore, unambiguous), and edges are undirected relations. The reason why directionality of edges can be ignored is that access should be bi-directional. At this stage we will also ignore possible names of relations (labels), in order to remain as general as possible, and to be able to consider the greatest number of resources.

2. **Completeness**: gives an estimation concerning the size of the lexicon (valued). If we aim at retrieving any word of a language, the resource should include all its words. This feature can only be evaluated in fuzzy terms, because no dictionary is or can be

\[\text{http://www.eat.rl.ac.uk}\]

\(^7\) Whenever we use the term ‘word’ we imply not only single terms but also ‘collocations’ or ‘multiword expressions’, that is, a sequence of words expressing meaning.

\(^8\) A word sense should be understood as an indexed word. For a word, there are as many indexes as the word form has senses.
complete, be it for reasons related to proper nouns (named entities), newly coined terms (neologisms), etc.

3. **Connectivity**: connected graph (constraint). This means that all words should be connected. The graph shall not contain any isolated nodes. This is an obvious property if our goal is to allow reaching a target from any input word.

4. **Density**: the average number of connections each node has with its neighbour nodes (valued). A small number would not be good. Let us take an extreme case. Imagine a lexicon in which words are ordered alphabetically, and the only connections from a word are towards its immediate neighbours, the previous and the next word in the dictionary. This arrangement defeats nearly all chances to find the target, as, in the worst case, one would have to traverse the whole lexicon in order to get from the source- to the target-word. At the other extreme are graphs with an extremely large number of connections. This is also undesirable. Imagine a totally connected lexicon, one where each word is linked to all the others (complete graph). This would not work neither. Even though the target word is always included in the list as direct neighbour, being surrounded by a great number of irrelevant words, it cannot be found in due time.

5. **Features**: each node of the graph should be characterised by a set of features, to be used later on for filtering and clustering (valued). Hence, once a word is spotted (step-1) it should spread activation to all its neighbours, possibly filtered according to some shared properties. This would yield clusters which still need to be named. Both operations are feature-based.

We present now a comparative analysis of some well-known resources along the lines just described.

**WordNet (WN)**

1. **Representation**: un-directed graph (passed). Vertexes here are synsets rather than words, and edges are relations (hypernymy/hyponymy/antonymy/etc.), ignoring their intrinsic directionality. Synsets are not only lists of equivalent words. Actually all words (or literals) being part of a synset have attached with them an index showing their word sense. For example the word “cop” is an element of 3 separate synsets, cop#1, cop#2 and cop#3, meaning respectively: policeman, to steal, take into custody. This being so it is easy to translate all synsets into their respective word senses.

2. **Completeness**: close to 100% for common words, but very low for proper nouns (at least for the Princeton WN version);

3. **Connectivity**: failed, because the lexicon is split in 4 isolated graphs: nouns, verbs, adjectives and adverbs (with very few cross-POS links: play, tennis). Nouns and verbs are connected internally, because of the hierarchy, but there is no guarantee for adjectives and adverbs;

4. **Density**: rather low, as the links correspond to the small set of relations dealt with in WN (about a dozen). Hence, every word sense displays only very few links;
5. **Features**: POS, LEMMA\(^9\), SENSE, but also sets of linked words as given by relations (even if they form also edges of the graph): HYPERNYMY-SET, HYPONYMY-SET, ANTONYMY-SET, etc.

Let us note that we could have a variant of this resource with nodes representing word forms (actually lemmas) rather than word senses. To this end one would clash all nodes representing different senses into a single node, merge the edges and operate a transformation on features. As such, the representation condition will pass too, the completeness will not change, and the connectivity will now be higher, because polysemous words will collect links from all its previous sense nodes. The POS and LEMMA features will not change (being identical in all previous nodes representing word senses), but SENSE will combine with each of the HYPERNYMY-SET, etc. (because the semantic relations characterise word senses and not words).

**Extended WN (Ext-WN)**

1. **Representation**: un-directed graph (passed, as above). In this resource, unlike in WN, the elements of a gloss are semantically disambiguated. Hence a gloss is likely to yield a rich set of links towards all the respective synsets;
2. **Completeness**: same as for WN;
3. **Connectivity**: perhaps passed, because the links added by the glosses allow to cross the POS barrier;
4. **Density**: higher than WN (apart from the traditional WN links, links generated by the glosses are added here);
5. **Features**: POS, LEMMA, SENSE, but also HYPERNYMY-SET, HYPONYMY-SET, ANTONYMY-SET, etc. (as the sets of words connected by the respective relations) and GLOSS-SET (word senses issued in the annotated glosses).

A similar variant as the one mentioned for WN, where nodes are words and not word senses, can be thought for Ext-WN.

**Edinburgh Association Thesaurus (EAT)**

1. **Representation**: undirected graph with nodes being words and edges associations (if considered in both directions) (passed);
2. **Completeness**: The EAT network contains 23,219 vertices (words) and 325,624 arcs (stimulus-response pairs). For what concerns us here, EAT contains about 56,000 words, which is much less than WN (perhaps not passed);
3. **Connectivity**: yet to be proved (the more associations are displayed for each source word, the higher the chance to obtain connectivity, but we are not aware of any empirical proof that a path can be drawn between any two words in this resource);
4. **Density**: medium;

---

\(^9\) Or sequence of lemmas, in case of collocations. Since, a node is here a word sense, its lemma should be considered as a feature.
5. **Features:** ASSOCIATION-SET (an explicit, direct link), POS (implicit, can be deduced). Here SENSE is not marked. Note that two other points that may count are the number of stimulus words and the number of responses produced to some input.

Note that there are quite a few other attempts to build word association lists. For example, the Free Association Thesaurus\(^\text{10}\) is probably the largest resource of this kind for American English. It produces 750,000 responses to 5,019 stimulus words. The goal of the ‘small world of words’ project\(^\text{11}\) is to build a multi-lingual map of the human lexicon. At present it contains more than five million responses for Dutch and more than a million for English. Given their size, these resources are quite likely to pass the test of completeness.

**An unstructured language corpus** (a representative collection of sentences of a language)

Let’s note that word occurrences in a corpus are distinct from words of the language (lexemes, or title words, as they appear in a dictionary or a thesaurus). In a corpus, the context could be exploited to build the connectivity, the context of the word occurrence \(w^{\text{occ}}\) being, for instance, all word occurrences (excepting for itself) belonging to the sentence to which \(w^{\text{occ}}\) belongs to. But, making a graph in which nodes are word occurrences yields a collection of small, disconnected graphs, ruining all chances for navigation. This is clearly an example of a bad representation for a resource like the one we need. To repair this, we need to connect the words belonging to different sentences to each other. One way of achieving this would be to integrate all occurrences of the same word into a single node.

1. **Representation:** an un-directed graph where nodes are considered to be words; Hence a word \(w\) has an edge towards another word \(w_j\) if the corpus contains a sentence with \(w\) and \(w_j\) appearing together (passed).
2. **Completeness:** complete if the corpus is large enough (passed);
3. **Connectivity:** passed, if the corpus is large enough;
4. **Density:** very high, perhaps even unmanageable for high frequency words, yet rather small for the rest. In accordance to Zipf's law, the density of the edges follows a power law distribution.
5. **Features:** POS, LEMMA, SENTENCE_ID X WORDS (this last feature is the vector product of the set of sentence IDs and a subset of all words; thus word occurrences are modelled by storing for each word/node a pair containing the ID of the sentence it belongs to and the list of all the other content words in that sentence). A variant of this resource can be a graph in which nodes represent word senses.

**Roget’s Thesaurus**

1. **Representation:** the graph could be thought of as a collection of isolated word entries (passed).

---

\(^{10}\) [http://web.usf.edu/FreeAssociation/Intro.html](http://web.usf.edu/FreeAssociation/Intro.html)

2. Completeness: complete\(^{12}\) (passed);
3. Connectivity: failed, as words are isolated;
4. Density: zero;
5. Features: CLASS, SECTION, SUB-SECTION, HEAD-GROUP, HEAD, POS, PARAGRAPH, SEMICOLON, LEMMA.

The representation, proposed above, does not support navigation. However, Roget’s Thesaurus has the merit to present a rich set of features attached to words and it is a good candidate to combinations.

Other resources could be taken into consideration: Wikipedia, when considered in combination with its hierarchy of categories, DBpedia, BabelNet, ConceptNet, etc.

The comparison of the resources along the various dimensions gives us an idea concerning their relative adequacy with respect to our goal. Note that we did not take into account the last property, called ‘features’. It seems that WN fails in terms of connectivity and density. Ext-WN escapes the connectivity problem, but its density is probably still too low to allow for a significant expansion in Step-1. EAT does not pass the completeness property and seems weak with respect to connectivity. The density of a corpus is likely to be extremely unbalanced, displaying either too many or too few links, depending on the word. Roget’s Thesaurus fails both in terms of connectivity and density.

To overcome the weaknesses of individual resources with respect to the TOT problem, combinations of resources could be considered. Obviously a combination of two resources is allowed if and only if the representation constrains with respect to nodes and edges are identical. This means that nodes should represent either words or word senses in both resources, in which case the combination will be formed by simply merging edges and combining features of identical nodes.

For instance, we could combine Ext-WN with EAT to overcome the density problem of Ext-WN. To this end we could combine the links generated by glosses with EAT’s associations. This would increase considerably the number of candidates at the end of Step-1. WN as well as Ext-WN seem to lend themselves well for the clustering operation referred to as Step-2. In both cases one could use the hypernymy links, at least for nouns: for instance, a group of words having the same hypernym (closest common ancestor) could be clustered together, while having already naturally a name, given by any member of the hypernym synset, or the whole set altogether\(^{13}\).

5. Outlook and conclusion

To summarize, we were dealing here with word access by people being in the production mode. Word finding is viewed as an interactive, fundamentally cognitive

\(^{12}\) Digitised by Jarmasz (2003), based on the 1987 version of Roget, published by Pearson Education.

\(^{13}\) Please note that the distance of the various elements with respect to a common hypernym may be quite variable, hence cluster names may vary considerably in terms of abstraction.
process. It is *interactive* as it involves two agents who cooperate (human/computer), and it is *cognitive* as it is based on knowledge. Since this latter is incomplete for both of them, they cooperate: neither of them alone can point to the target word, but working together they can. Having complementary knowledge they can help each other to find the elusive word.

How this can be accomplished precisely remains to be clarified in further work. Meanwhile we have sketched a formal representation of linguistic resources, on which a clustering and naming general strategy could be applied. While so far no single resource seems to be adequate to offer a satisfying solution, combining the right ones should yield a tool, allowing users to overcome the TOT-problem. While our ultimate goal is to help authors to *find* what they can’t *recall* based on whatever they can *remember*, at present we can offer only preliminary solutions. Clearly, a lot of work lies ahead of us.

**Acknowledgements**

Part of the work of the second author was done under the project *The Computational Representative Corpus of Contemporary Romanian Language*, a project of the Romanian Academy.

**References**