The Future of Linguistics and Lexicographers: Will there be Lexicographers in the year 3000 ?

Gregory Grefenstette

Gregory GREFENSTETTE, Xerox Research Centre Europe

The Future of Linguistics and Lexicographers: Will there be Lexicographers in the year 3000?

Abstract

Lexicography has been a respected tradition of scientific pursuit since the eighteenth century. Will the science of lexicography survive into the third millennium? I will discuss what present-day computational linguistics can offer the lexicographer in the lexicographic task. I will show that the data can be purified in clearer and clearer forms through approximate linguistics. I will then speculate about what aspects of the lexicographic task can be automated, what tasks remain incumbent on humans, and wonder whether a new vision of the lexicon will emerge.

Keywords: computational linguistics, lexicography, linguistics schools, parsing, lexical modeling

1. Introduction

Lexicology has been posed as a science ever since the 18th century. In the article *Grammaire* from Diderot and D'Alembert's *Encyclopedia*, lexicology is described as the explanation of three aspects of knowledge about each of the words from a language: its Material, its Value, and its Etymology. The Material is described as how a given word is put together: the syllables that compose it, how it is pronounced. The Value of a word is divided into three parts: the fundamental sense (proper or figurative), the specific sense (now called the part-of-speech), and the accidental sense (morphological variants of the word). The Etymology concerns the rules by which new words are formed, as well as the historical sources of a word interesting as an aid to understanding the word's current meaning. Lexicology dealt with words taken in isolation, while syntax concerned words in the context of other words. The section ends: "*Tels sont les points de vue fondamentaux auxquels on peut rapporter les principes de la Lexicologie. C'est aux dictionnaires de chaque langue à marquer sur chacun des mots qu'ils renferment, les décisions propres de l'usage, relatives à ces points de vue*".

Looking at modern dictionaries, we find that lexicographers in every language have taken these words to heart. The same principles are found in today's dictionary entries. The Material is presented in a word's spelling, pronunciation, and syllabification. The Etymology is often covered by a succinct presentation of the word's origins. Two of the three parts of the word's Value, the part-of-speech and special morphological variants, are almost invariably included in the entry. The remaining point of view on a word, a description of the fundamental sense, composes the meat of the dictionary entry, and, to push the culinary metaphor, is the bread-and-butter of the lexicographer.

In this paper I discuss what help the field of Computational Linguistics will bring to this task of describing the fundamental meaning of a word, and how the tools that will foreseeably be developed in this field may radically alter this two hundred year-old image of lexicographical description.

2. Linguistics, Computational Linguistics, and Approximate Linguistics

It is clear that one of the things that a computer can do well is treat a large amount of data, and that, with more and more text becoming available online, there is a lot of data with which a computer could work.[1] What is not clear is why Linguistics and, more recently and more blamably, Computational Linguistics have been so slow in providing adequate computer tools for aiding lexicographers in their task of describing word meaning.

Linguists used to be polyglots who were able to cite and qualitatively compare lexical and grammatical similarities and differences between languages. The systems that were devised in this old school of Linguistics mixed semantics, history, phonetics, into rules that were comprehensible to humans but not formally specifiable.

The idea of mathematical or computational linguistics appeared after the advent of the computer and following the mathematical classification of formal languages that Chomsky (1956) produced. There was a certain heady glory in the idea that man's languages could be reducible to mathematical principles, now implementable in such calculating machines as were only a dream in Pascal's time. This computational idea was new, exciting, and à propos as Universities grew and were restructured during the academic buildup of the 1960s. Purveyed by these new, young linguists, the idea that all languages were merely variations on a set of formal principles, and that discovery of the parametric settings for each principle might be determined, became the dominant theory in Linguistics and in Linguistic departments. Poor working lexicographers, with their down-to-earth questions about one particular language's vocabulary, were left in the dust as the linguists of the new generation pursued the more glorious goal of uniting the world's languages.

By the end of the 1980's two conflicting schools of computational linguistics were sharing the stage: the universal grammar school of Chomskian linguistics, and the no-grammar school of corpus linguistics. Here, if the reader permits, I will digress into a metaphorical comparison of these schools, inspired by Umberto Eco's comparison of the pre-Windows worlds of the DOS and of the Macintosh to Protestantism and Catholicism.[2] In my metaphor, one can map the Chomsky-inspired school to the communist ideology; and the corpus-based school, of which the late IBM research (Brown, 1992) was the most flamingly brilliant example, to pure capitalism.

The communist ideology is firmly anchored in the belief that there is a unifying logic underlying human life, a logic that can be understood, a logic of which one can tease the thesis and anti-thesis from objective historical events by the power of reason. Understanding this logic leads to perceiving the communist ideal, the final classless unity to which humankind is tending in its spiraling crises of production and revolution. In the same Utopian vein, we find the linguistic endeavor to understand the underlying parameters whose discovery will render the masses of languages simply variations of one universal language. The Chomskian linguist is out there, searching for truth among the conflict. Not concerned by the trivial, quotidian of languages, the Chomskian is looking for hidden currents, running through history and cultures, because their discovery will solve the divisions, and bring about the golden age. On the darker side, we find the same fanaticism concerning the unadulterated truth of the believer's path. Just as factions among the far left (and the far right) are violently

opposed to deviance from their party line, we find a plethora of acronyms GB, HPSG, CG, LFG, covering groups who ostracise and disparage their philosophical neighbors.[3]

The competing school of corpus linguists can be compared to pure capitalism in that they feel that the market (corpus) has its own self-defining logic (Adam Smith's *the invisible hand of God*) that should not be impeded by outside theories. Contrary to the Chomskian school, motivated by the goal of making all languages one, the corpus linguistic approach is undirected, taking an empirical approach of "seeing what is in the corpus". It is concerned with maximising what can profitably be extracted from the corpus, and in this sense, in order to increase its gains, its market must continually expand, encompassing ever larger corpora.

What is needed is a middle way between these two extremes of laissez-faire and linguistic *dirigisme*, that is, a recognition that there is some structure in text that can be recognised by machines, and that this recognition can be done in a useful way before the final goal of ultimate language unity is achieved. This middle linguistic ground is somewhat like the social-democratic approach of Western Europe: not theoretically pure, but containing a smattering of principles; recognising market forces, but ready to stick a political hand into the market. In lexicography, this approach has been attempted by a few computer scientists, such as Kenneth Church[4], who has taken the time to sit and watch eminent lexicographers at work, seeing what parts of their task might be readably computerisable, using a "combination of data and theory". Lexicographers such as Jeremy Clear (1993) have also explored this middle ground, attacking their real problems of lexicography, particularly the search for a description of words' fundamental meanings, with the computer.

This middle approach can be named *approximate linguistics*. Not perfect, but not bad. Until now, the use of computer parsers in lexicography has been considered limited by the unavailability of swift, robust parsers with which to process large corpora of naturally occurring texts. This dearth of practical parsers can be explained by the aptitude of linguists to consider that the inability to properly treat the least counter-example as a fundamental flaw in a parser, so, they consider, no parser can yet be proposed; it has also been sustained by the highly publicised belief in the all-computational camp that no parser is needed, since statistics can find any structure in a large enough corpus. Bridging these two extremes, approximate linguistics is based on the idea that higher linguistic analyses such as those to be produced by perfect full-sentence parsing, can be represented more or less faithfully by imperfect systems, that the accuracy of the approximations produced by these systems can be measured, and that the approximations can be successively refined by incremental improvements to the system to reduce error to desired proportions. In the next sections I illustrate an approximate linguistics approach to a lexicographical problem, showing both how the desired answers become clearer and clearer as more linguistic knowledge is injected into the system, and how one higher level linguistic analysis can be approximated by lower level functions.

3. Approximate Linguistics and Lexicography

One of the principal tasks of lexicography, as stated above, is to describe the fundamental sense of a given word. Currently, a lexicographer does this by seeing how the word is ordinarily used in the language, calling upon intuition or upon corpus citations (Kilgarriff, 1992:51). When a corpus is being used, the idea of ordinary use shifts somewhat, becoming quantifiable. How a word is ordinarily used in a corpus brings in the idea of frequency, how

often a word is used in a certain way, and the idea of relative frequency, how often it is being used one way as opposed to another. Now, having shifted the problem of fundamental meaning to ordinary meaning to a process involving counting, let us see how the computer enters the picture.

We can observe that the computer is very good at counting, but unfortunately not very good at knowing *what* to count. Contrary to what a small child can do, a computer can only count things which are made to look exactly alike. One programming task, then, involved in making language-related things countable by computer, is transforming them, abstracting away surface difference between two variant manifestations of the same phenomenon until the computer can see them as the same string of characters, of zeroes and ones in the same pattern. This is actually one of the aims of a complete parser: to produce a unique internal structure so that different sentences, even in different languages (Butt *et al.*, 1998), can be matched, compared, and counted by a computer. Until such parsers become widely available, the abstraction process provided by them can be approximated using existing technology involving lesser text processing means, by approximate linguistics. The successfulness of these approximations depends, as we will see, on the amount of linguistic information included in the technology.

3.1. Abstraction Levels and Approximate Linguistic Tools

If one thinks of the process of abstracting away surface differences in text as a linguistic process, then one can consider that there is a continuum of abstraction levels between the original text and the completely parsed version (whatever that might be) of a text. In the first steps of computer treatment of text, converting physically printed text into an electronic form, some meaning-carrying aspects of the text are regularly abstracted away. The meaning carried by differences in fonts, in type size, and in page layout often disappear in the electronic ASCII version of a text. For corpus analysis, this loss of information carries the advantage that the computer can now compare two strings of characters in an electronic corpus, rather than comparing pixel equalities, in order to determine whether the two words are the same. This first level of text abstraction requires linguistic information, either on the part of the person who retyped in the text and who was able to correctly recognise each letter, or by the optical character reading program, programs which commonly contain character-level language models.

Here are some other more evident linguistically-informed levels of abstraction which erase surface differences in words:

- tokenisation – deciding where to find the boundaries of the objects to be compared,
- lemmatisation – conflating inflected forms of words to normalised lemmas,
- part-of-speech tagging – abstracting away from individual words to grammatical classes of words,
- shallow parsing – abstracting away from positional information to syntactic function,
- semantic tagging – abstracting away from individual words to semantic classes.

Each of these tasks can be attacked more or less successfully with different computer tools. The tools used for one level can be stretched to perform tasks on a higher level. Just as when one only has one tool, like a screwdriver, one uses it as a hammer, a knife, or whatever is needed, so one can make do with inadapted computer tools to perform linguistic tasks. To illustrate what different levels of abstraction can provide to the lexicographer using current technology, let us consider a typical lexicographic problem. Suppose that the lexicographer

has a large corpus of English text, and is working on the fundamental meaning of the word *check*. More precisely, suppose that she is currently looking for the typical arguments of the verb *check*, an interesting example since the verbal and nominal forms of the word are written the same: to *write a check*, to *check*. Further suppose that, in order to grasp the ordinary senses for *check*, she is looking for the common direct objects (what can be *checked*) and the common objects of prepositional adjuncts (e.g., what can be *checked for*) of the verb *check*. If a full parser existed that could treat the entire corpus, one might then parse the corpus and the results of the parser would indicate these common arguments with precision. In the next few sections, I show what different levels of abstraction, made possible by incrementally more complicated linguistic tools, provide as an answer to this task. We shall see that simple tools provide an answer, but the answers become clearer as more linguistic knowledge is added, and as the approximations more accurately approach the goal of complete parsing.

3.2. Tokenisation as an Approximation to Parsing

In this task of finding common arguments of a word, tokenisation is the simplest tool that can be used to approximate parsing. Tokenisation defines the boundaries of the units that will be counted by the computer, and allows us to recognise contiguous but separate units. In printed languages which use spaces, the techniques of tokenisation usually involve using regular expressions to describe contexts where the input string needs to be separated into units (Karttunen *et al.*, 1996). Since tokenisation is a well known problem in computer science, being an inherent part of computer language compiler construction, many computer-based tools have been developed that can be used for tokenising, e.g. *lex, awk*, and perl.[5] A tokeniser for a natural human language contains simple linguistic information about how words are formed in that language: what characters are letters, what characters are numbers, what characters are punctuation, what elements of punctuation can appear within a word (like the apostrophe or the dash in English). The more linguistic information that the tokeniser has access to, the better the results (Grefenstette and Tapanainen, 1994).

As a first approach to our lexicographical problem, we can process our corpus in the following way: whenever the token *check* appears we store the next three tokens that occur in the input. This simple heuristic is based on the rudimentary linguistic knowledge that English verbal arguments and adjuncts generally occur shortly after the verb, so this three-token window is where we are likely to find the arguments we want. Once we have gone through the whole corpus, we count each occurrence of each token found in this small window, and then sort the tokens according to frequency. The computer is good at this counting and sorting. This same heuristic is similar to that used by practicing lexicographers visually scanning right-sorted KWIC (Key Word In Context) files for regularities. Table 1 shows the most common tokens extracted using this windowing technique for the token *check* over the British National Corpus.[6] (Here, even for this simple task, we can use different levels of approximation. The simplest tokeniser just splits words on spaces. The first three columns in Table 1 use this space-only approach. The last column uses a slightly more informed tokeniser integrating English word-structure knowledge to divide the input text into separate tokens.) The first column gives the most frequent tokens found in this three word window, with their frequency. This list is not very interesting for finding the direct objects and objects of prepositional adjuncts of *check*. We find *check it* and *check this* but this does not tell us very much about what the typical arguments of the verb *check* are. There is interesting information about *check* nonetheless: we find that the most likely prepositions associated with *check* are *on, of, out, with*, and that *check* seems to subcategorise for subclauses introduced by

that and *whether*. This is interesting, but not what we were looking for. One improvement to this simple approach is found in subsequent columns from which are removed a list of common function words, called stopwords in the Information Retrieval community. Stopword lists[7] are comprised of personal pronouns, articles, prepositions, conjunctions, etc., the closed-class function words of the language. For English such a list runs to about one hundred words. Of course, such a list can be counted as a slight injection of linguistic knowledge into the system. Filtering out such words provides a much cleaner list using this simple three-word window technique. Column two in Table 1 shows a more plausible list of common objects of *check*: *lists, things, progress, details, accuracy...*

Column three introduces a minor linguistic fact, i.e. that upper and lower case distinctions do not affect meaning inordinately in English, since such typographical variations can appear in headers, or in quoted speech. The input corpus for columns three and four has been transliterated into lower case, so we find more instances of both *check* as well as its possible arguments.

	Space-only		*No Stop Words*		*No Case Distinction*		*Tokeniser*
1753	the	55	list	65	list	92	list
722	that	24	make	30	carefully	40	carefully
620	on	23	things	29	local	34	progress
398	and	22	carefully	26	make	28	things
336	of	21	progress	24	things	28	local
317	out	21	local	24	details	28	details
281	with	20	details	21	progress	25	price
272	for	19	own	21	price	25	make
257	your	19	accuracy	21	erm	25	lists
242	it	18	blood	20	own	24	accuracy
217	to	16	new	20	new	21	facts
195	in	16	made	20	back	21	erm
185	is	16	back	19	lists	21	balance
180	a	15	time	19	accuracy	21	back
172	you	15	spelling	18	time	20	water
152	whether	15	facts	18	blood	20	spelling
131	if	14	sure	17	spelling	20	own
122	all	14	lists	17	made	20	number
120	their	14	information	17	level	20	new
116	.	14	erm	17	facts	20	made
112	up	14	ensure	17	ensure	19	information
112	his	13	water	16	water	18	action
104	this	11	validity	16	number	17	shirt
99	before	11	suit	15	sure	17	level
90	they	11	quality	15	information	17	increase
89	are	11	points	15	balance	17	ensure
84	at	11	list,	14	increase	17	doctor
81	what	11	doctor	14	doctor	17	blood
80	by	11	data	14	correct	16	work
79	I	11	actual	13	work	16	time

Table 1: Finding and counting the three words appearing after the token *check* in order to find the arguments of the verb *check*. Columns 1 to 3 use just spaces to find tokens. Column 4 uses a linguistically motivated tokeniser for English. Stopwords (closed-class words) are removed from columns 2 to 4. Columns 3 and 4 ignore case distinctions. As more linguistic information is added, the lists present the possible

arguments more cleanly and more completely. Note: *erm* comes from the recorded speech in the BNC.

The final refinement shown in Table 1 is the results of using a proper English tokeniser and not just spaces to separate words. We see that the counts for *checking a list* rises to 92 from column two's 55 count, since now we find cases where periods, commas, and parenthesis were stuck to *list*, effectively hiding it from our computer which could not recognise in column two that the two strings *list* and *list)* [with a postpended parenthesis] were the same word. What we see in the four columns of Table 1 are that little injections of linguistic knowledge make the lexicographer's task easier, by making surface differences disappear. These little improvements become especially evident as we go further down the list. For example, we see that *check .. shirt* appears whereas in the simpler approaches we would have had to scan further down the list to find it.

The results seem rather noisy, and require a certain patience and habit in order to extract any information. One can derive even from this noisy data that *details* and *prices* seem to be common arguments of *check*. But the confusion between nominal forms and verbal forms renders speculative any conjectures about whether *number* is a common argument of the verb *check* or whether it appears in a common noun phrase such as *check number*. In addition to confusing verbal and nominal forms of *check*, Table 1 ignores variant verbal forms of *check*. The next level of linguistic refinement, lemmatisation, addresses this problem.

checking		*checked*		*checks*		*check*		*check,-ed, -s,–ing*		*lemmas*	
17	watch	71	watch	58	balances	92	list	107	list	137	list
11	time	18	ensure	45	made	40	carefully	95	watch	118	make
10	records	17	hotel	21	carried	34	progress	67	made	98	watch
8	checking	16	carefully	7	performed	28	things	62	carefully	83	balance
7	stock	15	regularly	7	ensure	28	local	61	balances	62	carefully
7	progress	14	time	7	controls	28	details	49	make	57	number
7	erm	14	shirt	6	three	25	price	47	progress	51	ensure
7	back	14	checked	6	new	25	make	45	ensure	47	progress
6	times	13	number	6	make	25	lists	41	time	46	detail
6	quality	12	using	6	checks	24	accuracy	41	number	45	record
6	procedure	12	sure	6	built	21	facts	40	local	45	level
6	number	12	make	5	people	21	erm	37	things	44	shirt
6	make	11	list	5	gas	21	balance	37	back	44	price
6	local	9	three	5	files	21	back	36	erm	43	use
6	information	9	state	5	credit	20	water	36	carried	42	thing
6	facts	9	records	5	car	20	spelling	35	records	41	time
6	accounts	8	room	4	work	20	own	34	information	40	local
5	understanding	7	yesterday	4	traders	20	number	34	details	40	carry
5	system	7	times	4	take	20	new	34	accuracy	39	spelling
5	supplies	7	shirts	4	successful	20	made	33	new	38	car
5	purposes	7	rechecked	4	required	19	information	31	shirt	37	back
5	possible	7	pulse	4	regularly	18	action	31	facts	36	work
5	people	7	information	4	patients	17	shirt	29	regularly	36	erm
5	old	7	file	4	levels	17	level	28	water	34	information
5	notes	7	car	4	going	17	increase	28	sure	34	file
5	names	7	approved	4	data	17	ensure	28	price	34	accuracy

checking		*checked*		*checks*	*check*	*check,-ed, -s,–ing*		*lemmas*
5	movements	6	tables	4	cracks	17 doctor	28	lists	33	new
5	items	6	street	4	black	17 blood	28	car	33	fact
5	claims	6	statistical	4	appliances	16 work	27	times	32	name

Table 2: Approximating a lemmatiser by explicitly listing forms. Columns 1 to 4 show the tokens appearing within 3 words after various forms of *check*. Column 5 unions these lists. The last column gives the lemmas after any form of the lemma *check*. This last treatment finds more corpus samples.

3.3. Adding Morphological Analysis and Lemmatisation to the Approximation

A next level of linguistic sophistication, beyond tokenisers and lists of function words, is the ability to morphologically analyse and to lemmatise surface forms of words into some canonical form, for example, masculine singular for nouns, or an infinitive form for verbs. This requires the linguistic resources of a lexicon and an analyser, but thanks to the efforts of computational linguists and lexicographers (Karttunen, 1983; Chanod 1994) over recent years, these basic resources are becoming available in more and more languages.[8]

The last column of Table 2 shows the lemmas found after the lemmatised forms of the word *check*. We can approximate a lemmatiser, tediously, by explicitly detailing the forms we are looking are, such as are given in the first four columns of Table 2. The fifth column accumulates all four forms into one total column. The lemma-derived list in the sixth column is not all that very different at first glance from the fifth column, yet we have included a relatively expensive resource, a morphological analyser and lemmatiser in the process. In fact, the greatest difference comes in the numbers. Thus, in the same corpus, after lemmatisation, the number of recognised instances of *check ... spelling* grows from 27 to 41. This growth comes from the fact that the lemmatiser allows the computer to match variants such as *check my spellings* and *checked my spelling* to a single form *check ... spelling*, abstracting away morphological variation and intervening words within the window. The number of recognised instances of the desired phenomena grows as more linguistic information is added. Since the number of recognised instances grows, we can consider rarer phenomena. The morphological analysis and lemmatisation, by abstracting away differences, improves the counts of the data.

instances	*preposition*
1050	on
652	for
619	of
582	with
507	to

Table 3: Prepositions in a three word window after the lemma *check* in the British National Corpus.

Another advantage provided by a lemmatiser that returns the possible parts of speech as well as the lemmas, is that one can start to reason in terms of part-of-speech as well as in terms of strings. For example, one finds the most common prepositions in this window of three words after any form of the word *check* to be those given in Table 3. We will see the usefulness of this abstraction to part-of-speech as well as the improvement in recognition due to the abstraction derived from lemmatisation in the next section.

3.4. Adding Part-of-Speech Tagging to the Approximation

Table 2 shows a confusion between verbal and nominal uses of *check*. In order to distinguish these uses when the surface form is *check* used as a noun or *check* used as verb, we can supplement the tokeniser, morphological analyser and lemmatiser, with one additional linguistic tool: a part-of-speech disambiguator.[9] Part-of-speech taggers have been around for thirty years, based on hand-written rules, and statistics, attaining correct tag rates of 95% to 99% according to the languages (Karlsson, 1995; Schiller, 1996). Using such a tool is important for our problem since the identical nominal and verbal forms are both frequent. On top of the linguistic information contained in morphological analysers and lemmatisers about word forms, part-of-speech taggers contain models of word sequences to be found in the language. These models are either derived from introspection by linguistics, or from a training set of manually tagged text (Church, 1988). Using this linguistic tool allows us to further approximate perfect parsing, since now we can distinguish between verbal and nominal uses of *check* and look for arguments of the verb.

The first column of Table 4 shows the results of applying the three-word, window-based, parsing approximation technique to verbal uses of *check*, as well as to consider, if we wish, only nominal arguments to the verb. Here the solution to our original problem becomes even clearer. We see that one checks watches, the time, progress; details, lists, records, files; names, information, the car, levels, spelling, etc. From this list of potential arguments we see that some of the words appearing in the last column of Table 3, which came from a lemmatiser without a part-of-speech tagger, disappear. For example, *balance* is no longer present. This is because *balance* is almost exclusively found in the nominal expression *checks and balances*. The additional linguistic processing refines the search.

	Nouns after check...		*nouns found after check .. on ...*		*nouns found after check .. for ...*
97	watch	28	progress	21	sign
51	time	12	movement	14	error
44	number	11	thing	12	accuracy
43	progress	11	number	11	leak
37	record	9	file	11	damage
35	detail	8	use	10	check
33	thing	8	time	8	consistency
33	list	8	quality	7	time
32	accuracy	8	people	7	possible
31	fact	8	level	7	level
30	check	8	activity	7	correct
29	level	7	make	5	square
29	information	7	car	5	pulse
27	name	6	record	5	free
27	date	6	performance	5	flight
27	car	6	material	5	fingerprint
25	hotel	6	health	5	detail
23	work	6	gas	5	crack
23	spelling	6	child	4	wear
22	file	6	calculator	4	virus

Table 4: Once we can use a part-of-speech tagger with the morphological analyser and lemmatiser, rarer phenomena can be more accurately counted. We can specify what nouns appear after the verbal use of *check* in column 1. In Columns 2 and 3, we

can find nouns appearing after certain prepositions after verbal uses of *check*. This additional tool provides a closer approximation to full parsing.

This additional tool allows us to more accurately consider more specific and rarer phenomena. The second and third columns of Table 4 show the nouns appearing in a window of three words after the most common prepositions found in Table 3, *on* and *for*, themselves appearing within three words of a verbal use of *check*. From this, the lexicographer can be led to discover common uses involving prepositions: *check on progress/performance, check on car/gas/health/child,* or *check for damage/wear/crack,* etc. These samples are relatively rare in the corpus, and would have been further down the list in Tables 1 and 2 where they would have been swamped in noise, but here as the approximations to full linguistic analysis improve, they become evident.

3.5. Shallow Parsing as an Approximation to Full Parsing

As the previous sections have shown, some aspects of syntax can be approximated by simple position information, i.e. the window appearing after the word being examined. We have been supposing that words appearing in this window probably play some role as an argument. Other words appearing in the window are abstracted away so that different surface configurations can be made to look equal for the computer.

A further linguistic refinement that can be applied is to use regular patterns of the tags provided by the part-of-speech disambiguator to partially recreate the syntactic structure of the sentence. Recognising nominal chains and verbal chains as sequences of part-of-speech tags allows us to recognise certain syntactic relations, such as government of a noun by a preposition, or the voice of a verbal chain. This classification allows us to further pinpoint possible objects and prepositional arguments while eliminating others. This area of computational linguistics (Debili, 1982; Abney, 1991; Grefenstette, 1994; Ait-Moktar & Chanod 1997) is gaining more attention and more respect as much work progresses on creating such low-level parsers in a number of languages. Using such a low-level syntactic pattern extractor (Grefenstette, 1998) built using finite-state regular expressions and transducers, we can analyse the corpus at a higher level of abstraction.

	parsed *check ... with ...*		*3 words after* *check ... with ...*
20	office	19	local
8	doctor	19	doctor
7	authority	13	office
5	agent	12	level
4	number	9	spirit
4	manager	7	bank
4	detector	6	manager
4	company	6	erm
4	bank	6	company
3	staff	6	check
3	police	6	authority
3	hotel	5	travel
3	editor	5	police
3	consulate	5	agent
3	centre	4	time
3	boss	4	radio
3	association	4	parent

	parsed *check ... with ...*		*3 words after* *check ... with ...*
2	time	4	own
2	thermometer	4	manufacturer
2	test	4	language

Table 5: Comparing results returned with a shallow parser to those returned by a window-based approximation to parsing. The first column gives the objects of the preposition.

shops and other establishments abroad	check	with your bank for details .
So it is essential to	check	with your travel agent or bank on
No, cheque - but it 's good - Josh	checked	with the bank, called Hnatiuk
this was a good idea but would have to	check	with his immediate boss
Universe editor Anne Noels	checked	with bosses about the ban in
It is sensible, however, to	check	with editors of really specialist
On being told by the manager to	check	with the bank, he pretended to
Then, posing as a relative, she	checked	with the editor .
She	checked	with all the contributing editors
with drivers stranded in France	checking	in with their boss .

Table 6: KWIC lines associated with *check .. with (bank or editor or boss)* lines returned by the shallow parser.

Table 5 shows the improvement in the data that the shallow parser tool gives over the lower-level 3-word window approximation for the pattern *check...with....* As before, the lists become cleaner as noise becomes more properly eliminated. One can see, in the first column, that in the corpus people *check with* their bosses, editors, and banks. This shallow parsing information can be used to extract the corresponding KWIC lines automatically for the lexicographer, as shown in Table 6.

80	watch	92	list
44	number	84	check
33	record	40	carefully
33	detail	34	progress
30	time	28	things
30	item	28	local
30	fact	28	details
29	level	25	price
29	date	25	make
27	accuracy	25	lists
24	list	24	accuracy
22	work	21	facts
...	...	...	...
17	condition	17	ensure
17	answer	17	doctor
16	story	17	blood
16	room	16	work
...	...	...	...
4	symptom	5	vehicle
4	structure	5	various
4	string	5	use
4	status	5	ups

4	slide	5	trading
4	sign	5	traders
4	shoe	5	table
4	setting	5	supplier
4	security	5	suitability
4	seal	5	still

Table 7: Parsing vs. Window. Looking for direct object collocates of *check*. The first column shows nouns extracted as direct objects by a low-level parser, the second column shows words within a window of three words after *check*. In the first column, we see that noise coming from prepositional objects (*check with doctor*), and noise coming from non-recognition of structure (*check your blood pressure*) which clutter the list supplied with the simpler approximation, disappears from the first column. Even much further down the list, one can capture rarer but true arguments more often with the shallow parser.

Table 7 shows the lemmas identified as direct objects that are extracted using a low-level parser over the same BNC corpus used throughout this presentation. Though noise persists, as is the case with all the previous approximations to full linguistic parsing, the lists produced are more precise. In both columns of Table 7, there is much overlap with the most frequent cases, but the focusing power of adding more linguistic knowledge appears more clearly at the end of the list where one discovers only with the low-level parser that one *can check symptoms, signs, shoes, security*. These lemmas would also appear further down the list of the second column but they would be swamped in the noise there and be harder to discern.

The initial problem of finding argument for the verb *check* has been treated with a sequence of increasingly sophisticated linguistic tools: from tokenisers, to eliminating a list of stopwords, to morphological analysers and lemmatisers, to part-of-speech disambiguators, and finally to low-level parsers. The results obtained in Tables 1 to 7 show a gradual focusing in which more noise is eliminated and in which more infrequent phenomena are brought to light as more linguistic information is incorporated into the process.

3.6. Semantic Tags

One further linguistic refinement, this time using the WordNet (Miller, 1990) thesaurus as a linguistic resource, is shown in Table 8. This table shows the semantic tags associated with each of the most frequent direct objects recognised for *check*. Semantic tags are not as well defined as grammatical tags where the classes are more constrained and better understood (Bolinger, 1965). Ideally, a semantic part-of-speech tagger would use context, as a grammatical part-of-speech tagger does, in order to choose the most likely tag. This research area has had limited success (Segond *et al.*, 1997), impeded by a lack of semantic dictionaries and due to the fact that the problem is no longer one of simple structure but also of meaning.

The only conclusion to be drawn from Table 8 is that the direct object of *check* is likely to be something classified as a *communication* or an *artifact*, but the meanings of these semantic tags are not very clear in themselves.

	direct object of check	*WordNet semantic tags associated with word*
80	watch	act, artifact, time
44	number	attribute, artifact, communication, group, quantity
33	record	act, communication, possession, quantity
33	detail	cognition, communication, group, relation
30	time	Tops, event, time
30	item	artifact, communication
30	fact	cognition, state, communication
29	level	attribute, artifact
29	date	person, group, food, time
27	accuracy	attribute
24	list	communication
22	work	act, artifact, phenomenon,
22	thing	attribute, act, artifact, cognition, communication, event, feeling, state
22	spelling	communication
22	position	act, attribute, cognition, location
22	name	communication, person, group, state
22	file	artifact, communication, group
20	shirt	artifact
20	result	event
19	pressure	attribute, cognition, phenomenon
18	progress	act, event
18	figure	act, attribute, artifact, cognition, shape, possession, communication
18	car	artifact
17	price	attribute, communication, possession
17	information	cognition, communication
17	equipment	artifact
17	condition	communication, state
17	answer	act, communication
16	story	communication
16	room	artifact, group, quantity, state

Table 8: The WordNet semantic tags associated with the most common direct objects of a verbal form of *check*. The semantic tag *communication* (16 of 31 words) or *artifact* (12 of 31) appear most frequently. Choosing between the semantic tags using context is a problem akin to part-of-speech disambiguation, but on a much larger scale, as the ambiguity of words is greater than with grammatical tags, and because there are many more potential semantic tags than grammatical ones.

4. Extrapolation to the Future

So far, I have presented the historical argument that the main task of the lexicographer is to describe the usual meanings of the words in a language. I have also argued that classical linguistics (1956-present) has not been supplying the tools necessary for this task because it has been concentrating on the universal (universal grammar formalisms or universal statistical techniques) and not on the everyday routine. Springing up, and gaining weight in computational linguistics, are techniques that can be called approximate linguistics, i.e., approximations to linguistic theory that are both incrementally perfectible as well as being robust and immediately useful.

The previous section gives a hint of what approximate linguistics can offer working lexicographers today in their task of describing some fundamental sense of a word. Though the tools used there have not all been packaged as a shrink-wrapped product with lively colours and series of ergonomic pull-down menus, all the tools exist today from a variety of resource providers and in a variety of languages: corpora, tokenisers, morphological analysers and lemmatisers, part-of-speech taggers, shallow parsers, and assorted bookkeeping tools which count, sort, calculate co-occurrence statistics (mutual information, t-scores, etc.), and so on. These tools allow the lexicographer to see what words co-occur with a given word, to examine what syntactic structures the word occurs in, what other words enter into what syntactic relations with the word and how often, and to retrieve all the corpus lines corresponding to any of these configurations.

Let us return to the question of corpus size, which is one of the major aspects of the theory-less corpus linguistics approach. All the tables and charts in this paper were derived from the 100-million word British National Corpus. In this large corpus, the lemma *check* appears about 13,000 times. This great number of occurrences allows us to aid our imaginary lexicographer by showing the common arguments of the word *check* as a verb. With little linguistic information, e.g. stopwords, we already get some answers; as more linguistic information is added to the system, the results become clearer, and rarer phenomena can be considered, as can be seen at the bottom of the first column of Table 7. As an example of how more complex structures become rarer, in our Xerox-parsed version of the BNC, there are 4854 instances of *check* with a direct object; when we consider prepositional adjuncts, a structure that involves three words: the verb *check*, a preposition, and the object of the preposition, there are 2893 cases; when we consider the more specific structure of prepositional adjuncts to *check* beginning with the preposition *with*, there are only 417 cases (see Table 5). In true Zipfian fashion, the more complicated the structure, the rarer it is. The rarer something is, the more corpus you need to find it reliably. Now, on the Altavista Web browser in Spring of 1998, there were 13 million indexed occurrences of a form of *check*, one thousand times more data than was used for the data appearing in this presentation.
The data is there for extracting all the information one might want about a word in more and more complicated patterns.

Will we need lexicographers in the future? What are lexicographers needed for now? According the Adam Kilgarriff (1992), the ideal lexicographer must (i) gather corpus of citations for a given word, (ii) divide the citations into clusters, (iii) decide why the cluster member belong together, and (iv) code their conclusions into a dictionary definition. As for the first two steps, we can easily predict that approximations to proper parsing that computational linguistics can provide will become more and more accurate, that the structures that will be recognised will be more and more complex, that the corpus to which the techniques will have access will be bigger and bigger, and that more and more usage patterns of words can be extracted from this text corpus. The computer will have better numbers to count with, to cluster with, to separate with. These leaves the rational steps of three and four, synthesising and explaining. Synthesising means finding a higher level description of the things found in certain positions. Table 8, the WordNet semantic abstractions, shows that this goal is far from being satisfied. It is something that humans are eminently good at, but machines still poor. Explaining what makes a cluster coherent in a way understandable for a human is beyond the scope of a computer. It means drawing distinctions and contrasts between shared experiences and expectations, explaining what makes this group

different from other groups that the human user knows. In this sense, that of providing shortcuts to understanding to other users, a lexicographer will always be needed.

Yet it might also happen that lexicographers may be needed less and less for some of the tasks that they spend much time on now. What if end-users had access to clustered and filtered KWIC lines when they wondered how a word was used? What if the computer was able to analyse any word in context, extract all the patterns that the word was found in, cluster them in a coherent way[10], and display these further examples succinctly for the user? For translation dictionaries, for knowing the patterns that words appear in, with what preposition, what if patterns were able to be pre-digested and scanned by the user, could they induce meaning from these examples? Would the user need a human-supplied explanation? I believe that in the coming years we shall see a new way at looking at the lexicon that is no longer two-dimensional: a list of words, and their explanations, but rather three and four and five dimensional in which information is stored about how each word is used with each other word, and how that pair of words is used with a third word, and that triple with a fourth word. This vision requires massive data structures and robust linguistic-approximation tools to fill them, and a new way of sifting and handling this data. It also requires a lot of disk space, but this, contrary to the space requirements that governed paper dictionaries, is now the least of our concerns when we remain in the electronic world.

But paper still has a long life in front of itself, and as long as things are printed we will need the reasoned condensations that only lexicographers provide.

5. Notes

1 For example, the Altavista web browser has encountered the word *the* 1.4 billion times in its latest monthly trawl of the Internet. With its frequency of about 1 per 7 English words, we can roughly estimate that there are 10 billion English words to be computationally digested on the Web pages accessible through Altavista. The words are certainly there, accessible for lexicographical study, in this large distributed corpus.

2 See http://www.hut.fi/~tsiivola/Eco-Mac_vs_Dos.html. Here is an excerpt: "The fact is that the world is divided between users of the Macintosh computer and users of MS-DOS compatible computers. I am firmly of the opinion that the Macintosh is Catholic and that DOS is Protestant. Indeed, the Macintosh is counter-reformist and has been influenced by the 'ratio studiorum' of the Jesuits. It is cheerful, friendly, conciliatory, it tells the faithful how they must proceed step by step to reach – if not the Kingdom of Heaven – the moment in which their document is printed. It is catechistic: the essence of revelation is dealt with via simple formulae and sumptuous icons. Everyone has a right to salvation. DOS is Protestant, or even Calvinistic. It allows free interpretation of scripture, demands difficult personal decisions, imposes a subtle hermeneutics upon the user, and takes for granted the idea that not all can reach salvation. To make the system work you need to interpret the program yourself: a long way from the baroque community of revelers, the user is closed within the loneliness of his own inner torment."

3 Ivan Sag raged against the hubris of the Chomskians in a 1993 interview (http://www.coli.uni-sb.de/~mineur/Itv/Sag.html) in which he bemoaned that GB (Government & Binding) practitioners "can simply condemn a whole tradition to irrelevance, like Relational Grammar, whose systematisations of complex grammatical systems constitute to my mind a long-lasting and significant contribution to the field of syntax. It is an anti-intellectual attitude, and in fact I think it is dangerous for the entire field of linguistics." He complained that GB "also has heresy - essentially what everyone else in linguistics does, not to mention the work on language done in other fields, e.g. psychology or computer science." Bob Carpenter, in a 1995 interview (http://www.coli.uni-sb.de/~mineur/Itv/Carpenter.html), describes the theoretical correctness that this school of Linguistics searches for: "I really hated HPSG (Pollard *et*

al., 1987) at that time, because to me it looked like a complete Frankenstein theory. Not because it is built up from all these other grammar formalisms, like LFG and GPSG and CG and all that, but I had a real bias against the fact that it was using different sorts of mechanisms to do everything. You can look at Categorial Grammar, and you can say there is a pure logical reasoning mechanism on which everything is based. But in HPSG that uniformity comes only at the level of the feature structure logic. It is important not to confuse the kind of underlying attribute-value logic formalism with the linguistic theory. In HPSG there is this whole linguistic theory that contains things like the Head Feature Principle and the Quantifier Binding Condition and Principle A of the Binding Theory. That is the linguistic theory. HPSG should not be criticised on the fact that it is based on a Turing-powerful constraint resolution system. You just cannot do in the linguistic theory, everything you can do in the underlying theory. Shieber originally pointed this out in the context of PATR." He continues with a description of how none of the theories propounded in this linguistic school are aimed at producing anything practical for NLP: "Now of course no linguistic theory, GB, HPSG, any of these, have really laid out a meta-theory of what really counts as a grammar. You will never find anybody saying: 'A GB grammar is the following thing: one of these versions of this, one of these versions of that.' They talk about it that way, they say it is somehow parameterised, but you never actually see a list of parameters. No one can make a proposal so that someone will stand up in the audience and say: 'Ah, but that is not a GB grammar!' Similarly, I could add a bunch of features and devices to HPSG and no one could say: 'That is not HPSG.' You can say that it is not in the spirit of HPSG, or not in the spirit of GB, but it is really all just a matter of esthetics. So I think insofar as people really want to make theories of a universal grammar, then they should be honest and lay out the possible grammars. Again, GPSG perhaps came the closest to this goal in spirit, but it didn't really excite many linguists. Once you lay something out concretely, it's just a little too easy to see where it falls down."

4 See the description of his invited talk to the European Association of Computational Linguistics in 1993 at http://www.coli.uni-sb.de/~mineur/Itv/Church.html.

5 See the following Web sites for free versions of these tools: http://www.cs. columbia.edu/~royr/tools.html, http://w4.lns.cornell.edu/public/compdoc/info/gawk/gawk_1, http://www-cgi.cs.cmu.edu/cgi-bin/perl-man.

6 The British National Corpus (BNC) is a 100 million word collection of samples of written and spoken language from a wide range of sources, designed to represent a wide cross-section of current British English, both spoken and written. See http://info.ox.ac.uk:80/bnc/. A version of this corpus, retagged using Xerox taggers (http://www.xrce.xerox.com/research/mltt/Tools/pos.html) was used for this paper.

7 Available via ftp in the directory /pub/med/smart/ at ftp.cs.cornell.edu.

8 See http://www.xrce.xerox.com/research/mltt/Tools/morph.html for online morphological analysers for most Western European languages.

9 The Common Lisp source for version 1.2 of the Xerox part-of-speech tagger is available for anonymous FTP from parcftp.xerox.com in the file pub/tagger/tagger-1-2.tar.Z. Another freely available English tagger, developed by Eric Brill http://www.cs.jhu.edu/~brill, uses rules based on surface strings and tags.

10 See the lexicons derived by the DECIDE project at http://engdep1.philo.ulg.ac.be/decide/lexicon and the reference Grefenstette *et al.* (1996).

6. References

Abney, S. 1991. "Parsing by Chunks" in Berwick, R., Abney, S., & Tenny, C. (eds) *Principle-Based Parsing.* Dordrecht: Kluwer.

Ait-Mokhtar, S. & Chanod, J.P. 1997. "Incremental Finite-State Parsing", *ANLP'97*, Washington, pp. 72-9.

Bolinger. D. 1965. "The Atomization of Meaning," *Language* 41(4), pp. 555-473.

Brown, P. F., Della Pietra, V. J., deSouza, P.V., Lai, J.C. and Mercer, R.L. 1992. "Class-based n-gram models of Natural Language," in *Computational Linguistics*, 4(18), pp. 467-479.

Butt, M., King, T., and Segond, F. (to appear). *A Grammar Writer's Handbook.* Stanford CSLI Publications..

Chanod, J.P. 1994. "Finite state composition of French verb morphology." Technical Report MLTT-005. Xerox Research Centre Europe, Meylan, France, http://www.xrce.xerox.com/publis/mltt/mltt-005.ps

Chomsky, N. 1956. "Three models for the description of language," *in IRE Trans. on Information Theory* IT- 2:3, pp. 113-124.

Church, K.W. 1988. "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text," in *Proceedings of the 2nd Conference on Applied Natural Language Processing,* Austin, pp. 136-143.

Clear, J. 1993. "I can't See the Sense in Large Corpus," in F. Kiefer, G. Kiss, and J. Pajzs (eds.) *Papers in Computational Lexicography COMPLEX'94,* Research Institute for Linguistics, Hungarian Academy of Sciences, Budapest, pp. 33-47.

Debili. F. 1982. *Analyse Syntaxico-Sémantique Fondée sur une Acquisition Automatique de Relations Lexicales-Sémantiques* Phd, University of Paris XI.

Grefenstette, G. *Explorations in Automatic Thesaurus Discovery.* Boston: Kluwer. Chap. 3.

Grefenstette, G. and Tapanainen, P. 1994. "What is a word, what is a sentence? Problems of tokenization." in F. Kiefer, G. Kiss, and J. Pajzs (eds.) *Papers in Computational Lexicography COMPLEX'94,* Research Institute for Linguistics, Hungarian Academy of Sciences, Budapest. Available at http://www.xrce.xerrox.com/publis/mltt/reports/mltt-004.ps.

Grefenstette, G., Heid, U., Schulze, B.M., Fontenelle, T. & Gerardy, C., 1996. "The DECIDE project: Multilingual Collocation Extraction." *Seventh Euralex International Congress*, Gothenburg, Sweden, August 13 – 18, 1996.

Grefenstette, G. 1998. "Light parsing as finite state filtering." in A. Kornai (ed) *Extended Finite State Models of Language*, Cambrisge Univeristy Press.

Karlsson, F., Voutilainen, A., Heikkilä, J., & Anttila, A. editors. 1995. *Constraint Grammar : a language-independent system for parsing unrestricted text*, volume 4 of *Natural Language Processing*. Mouton de Gruyter, Berlin and New York. 430 pp.

Karttunen, L. 1983. "KIMMO: a general morphological processor," in *Texas Linguistics Forum.*

Karttunen, L., Chanod, J.P., Grefenstette, G. and A. Schiller. 1996 "Regular Expressions for Language Engineering" in *Natural Language Engineering*, 2(4), pp. 305-328.

Kilgarriff, A. 1992. *Polysemy* PhD Thesis, CSRP 261, University of Sussex p. 51. Available via ftp at ftp://ftp.cogs.susx.ac.uk/pub/reports/csrp/csrp261.ps.Z

Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., and. Miller. K.J. 1990. "Introduction to WordNet: An on-line lexical database." *International Journal of Lexicography*, 3(4), pp. 235-244.

Schiller, A. 1996. "Multilingual Part-of-Speech Tagging and Noun Phrase Mark-up", in *the 15th European Conference on Grammar and Lexicon of Romance Languages*, Munich, Sept. 19-21.

Segond, F., Schiller, A., Grefenstette, G. & Chanod, J.P. 1997 "An Experiment in Semantic Tagging Using Hidden Markov Model Tagging." *ACL'97 Workshop on Information Extraction and the Building of Lexical Semantic Resources for NLP Applications*, Madrid, July 7-12.