A Cornish to English MT System

Paul R. Bowden
School of Computing and Informatics
Nottingham Trent University
Nottingham, England
NG1 4BU
paul.bowden@ntu.ac.uk

Abstract

A direct machine translation (MT) system for Cornish (Kernewek) to English is described. The *kern* system is still being developed but is already effective as a HAMT (human-aided machine translation) tool. The system first pre-processes the source Cornish text to replace single-sense multi-word phrases with their English equivalents. Word-for-word transfer then follows, using a lexicon that includes all the English senses for each Cornish orthographic word. Several post-processing stages then occur, such as forming *-ing* participles, *this/that* detection, and disambiguation of certain closed-class Cornish words. It is planned that general word-sense disambiguation (WSD) will then occur in order to resolve any remaining multi-sense words down to a single sense, but this has not yet been implemented. This WSD stage will take place on the English side, without recourse to the source text, so that it can be used for other language pairs. Higher-level structural rearrangement will also take place eventually; only a couple of simple structural transformations are currently done. The output from *kern* is already understandable to an English-speaking reader prepared to select from any multiple word senses still left. The Kernewek Kemmyn orthography is used; the growing lexicon currently holds about 9,000 Cornish headwords, with mutated forms held as separate entries. The program is written in 'C' and runs under LINUX/UNIX; it is console-based, prompting the user as necessary; an assisted interactive mode is provided to add unknown Cornish words to the lexicon "on the fly".

Introduction

Kernewek (Cornish) is a revived P-Celtic ("brythonic") language which has recently increased in importance in the UK after its recognition under Part II (Article 7) of the European Charter for Regional and Minority Languages, in November 2002 (Cornwall County Council, 2004). Despite a growing interest in and enthusiasm for the language, both in the region where it was once widely spoken (Kernow, the current Duchy and County of Cornwall, in the extreme south-west of mainland Britain) and outside this area, there are few language tools (LT) currently available for Kernewek. This lack of LT applies to both lexical and processing aspects; there appear to be no readily available large machine-readable lexica, no serious corpora of Cornish texts, and very few programs such as grammar checkers, spellcheckers, parsers and MT systems. The author is aware of one spellchecker/grammar-checker/corpus-builder in the pipeline, based upon the work of Scannell and Werner (website as referenced), and a single experimental

parser (Tyack, 2004) which is capable of handling only simple sentences. There has also been what appears to be an initial attempt at a tagset for Cornish (LER-BIML Project, University of Lancaster) but it is not clear to what uses these word-tags have been put, if any. Furthermore, as far as the author is aware, there is not yet available any operating system in Cornish (Windows or terminal-based).

Cornish today has more than one spelling system (orthography), and that used by the program described in this paper is Kernewek Kemmyn (Common Cornish), a relatively recent spelling system suggested initially by George, who produced new dictionaries (e.g. *An Gerlyver Kres* – George, 1998) and which was then adopted by Kesva An Taves Kernewek (The Cornish Language Board). This has become the most-used spelling system today[1], at least in terms of numbers of speakers and learners. There have been small-scale attempts in the past at MT from English to Cornish and *vice versa* in other Cornish orthographies but none of these systems had at their disposal large lexicons or indeed anything like full grammatical processing, and thus often gave disappointing results[2]. On the other hand, there are thriving websites for each orthography (e.g. *Nowodhow Kernow* and *Warlinenn*), for learners (e.g. the BBC's Cornish language pages) and even bilingual online shopping sites (e.g. *Just Cornish*). Paper-based resources such as textbooks for learning the language (e.g. Brown, 1996 and 2001) and magazines (e.g. *An Gannas* – "The Messenger", *pub*. Kowethas an Yeth Kernewek) are available, but computerised LTs are very rare. In this era of growing awareness of and interest in the Cornish language, and the use of computers in all aspects of modern life, the time has undoubtedly come to start creating serious Kernewek LTs, for day-to-day computer users, for Cornish-language learners, for Internet users, for government office staff, and for linguists.

The *kern* system being developed by the author is a step in this direction. It is a mono-directional system at present (Cornish to English only) designed to act, at least initially, as a HAMT (human-aided machine translation) tool, eventually to be made available in a number of environments. Written in the 'C' programming language and running currently under LINUX/UNIX operating systems, it uses a simple prompt-based interface, adequate during system development. A major part of the development work involves the design and construction of the lexicon, described in-part in an earlier paper (Bowden, 2003), and representing a valuable LT resource in its own right. This paper describes the current state of the *kern* program and its lexicon.

Design Motivation and Approach

Two desires have motivated the creation of the *kern* system. Firstly, there is the obvious

1 The other major orthography-cluster is Unified/Unified-Revised; it is only correct to inform the reader that not all Cornish revivalists view the change from this system to Kernewek Kemmyn as a good thing, and indeed arguments over which spelling system to use have sometimes become heated, undoubtedly to the detriment of the Cornish language revival movement.

2 At the time of writing, a search for any usable MT system for Cornish on the Internet proved fruitless, although there was a suggestion that one English-UC system did once have a web presence.

desire to create a working MT system that can be used by non Cornish speakers, and learners of the language, to translate Kernewek texts into English. Clearly, FAHQT (fully automatic high-quality translation) is desirable, but in practice this result is highly unlikely due to the essential complexity of the MT task (see e.g. Hutchins and Somers, 1992). However, the more realistic situation of a system that reveals the meaning of the source text, in not quite fluent English, would be acceptable. This system would be a MAHT (machine-aided human translation) or a HAMT (human-aided machine translation) system, depending on how it was being used, or indeed how successful it was. The *kern* system has, in fact, already reached the point at which a human (who doesn't speak any Cornish) could understand an input Cornish document, albeit with the application of a little effort to pick the correct word senses from the English output where more than one occurred, and also to supply missing meanings to the lexicon when prompted to do so. However, this latter task is done in an interactive mode designed for those having some knowledge of Cornish grammar and vocabulary. As the lexicon grows, the incidence of unknown words is dropping; it will soon be the case that unknown words are mostly inflected or mutated[3] forms of existing headwords, rather than being completely new headwords in their own right.

The second motivating factor in the creation of *kern* is to prove a point (or otherwise). The *kern* program follows (mostly) a largely out-of-favour MT methodology, known usually as "direct MT". This was the earliest form of MT, tried by the pioneers of MT in the early days of the computer, e.g. in the 1960s . The direct approach uses word-for-word transliteration, followed by post-processing stages to "tidy up" the output from the first stage. One of the biggest problems encountered by early researchers using this approach was that of WSD (word-sense disambiguation). Many Cornish orthographic words (i.e. strings of characters) can have several meanings (senses) – so *which* English word should replace the Cornish word? Another set of problems, as discussed by (Somers, 2004), relates to the need to perform phrase- and sentence-structural rearrangement. For example, at the phrase level, *hel an dre* means literally "hall the town", i.e. "town hall", and *yma edhomm dhymm* means literally "there is need to me" i.e. "I need". This latter example essentially involves a nominal to verbal transformation. However, as the set of such transformations is quite small ("there is hunger to me", "there is fear to you(s)" etc.), it seems feasible to write simple individual functions to handle these. At the sentence level, although Cornish does use SVO sentential order (subject, verb, object) just like English, it also uses VSO and other constructions under certain circumstances: e.g. *Yma Iowann ow tonsya* means literally "He/there is John dancing" i.e. "John is dancing". In order to resolve such constructs, the *kern* post-processor will undoubtedly have to identify noun phrases (NP).

3 *Mutation* is a feature of Celtic languages: the initial letters of words may change or disappear, dependent on the preceding word. Mutation systems can be complex – for example, in Cornish there are five "states" of mutation (including the dictionary form of the word) and a complex set of rules specifying which individual words or categories of words cause mutation in the following word, and to which states. See e.g. *Skeul An Yeth Vol. 1* for a mutation table. Cornish dictionaries do not usually list mutated words as separate entries, which makes it difficult for learners to look up words which may or may not be mutated.

These sorts of problems, together with the emergence of new and apparently more successful MT approaches based upon statistical and memory based methods, led to the demise of the direct approach in most new MT systems. However, the author believes that the direct approach should not be so quickly written off, especially for minority languages where there may not be many texts available, and the *kern* project will attempt to prove that this approach, or at least an augmented direct approach, can indeed be successful.

Concerning the Lexicon

The lexicon in the *kern* system is both machine-readable and human-readable. The initial construction of the lexicon has been described in (Bowden, 2003) but since then it has grown in size to over 8,000 entries and its entry format has been more tightly defined. The lexicon is in itself a useful resource, for NLP programs and also for learners of Cornish. However, it is *not* a dictionary. Its main purpose is as a LT resource, and it is a happy circumstance that it is also useful to human readers. A lexicon entry comprises a Cornish word, an equals sign with a blank either side, and a set of English senses (meanings) separated by forward-slashes if there is more than one of these. No attempt at explaining the senses is made – they are just strings of characters. For example, here is a lexicon entry:

res = necessity/race/row/(vbl_ptl_re)/races(3,s)/race!(2,s)

Clearly there are conventions used in the format of lexicon entries. These have been described in the *gerva.README* document which relates to the lexicon (available from the author). (A section of the lexicon is given in Appendix A.) The conventions detail the order of parts of speech in the English side, verb conventions used, etc. It is worth noting that, at present, there are types of information missing from the lexicon entries, such as parts of speech (PoS) (although verbs can be recognised from tags such as "(3,s)"). This is a deliberate design position. The philosophy behind the lexicon is to provide *only* the information *needed* to create the translation. No other factor matters – no information is given, for example, to help a human reader decide the sense or PoS of the Cornish word. Thus *res* means (amongst other things) "race" - but whether this means the sort one runs in, or the sort one belongs to, is not revealed by the lexicon entry. The point is that it will eventually be revealed in the *translated* sentence – by context. As far as the initial stages of the *kern* processing are concerned, the string on the right of the equals sign is merely what must replace the word *res* in the input text. To repeat, therefore: the lexicon is not designed to be a dictionary for humans.

However, since it does turn out to be useful – for example, mutated forms have their own headwords, which is very useful to a beginning learner of Cornish – then a "reverser" program has also been developed by the author. This splits the English senses into separate headwords, and assembles an English to Cornish version of the lexicon. Where more than one Cornish word can be attached to the English headword, this is done with slash separators. A section of this is given in Appendix B. The reversed lexicon is useful for reminding oneself of a Cornish word given the English, but of course there is no explanation. There are also many entries starting "to_" which come from verb infinitives. Furthermore, since a basic tenet of the lexicon is that *where there is more than one closely-related English sense to a Cornish word, only one is used* then the reversed lexicon is not like a dictionary – one cannot expect there to be a full set of near-synonyms present in the reversed list. For example, the lexicon entry for *nija* ("to_fly") does not also have attached meanings "to_soar", "to_glide" etc, and since these senses do not currently

have their own Cornish words, they are not present in the reversed lexicon as headwords.

The lexicon is available free to anyone who requests it, together with the README file that explains the conventions used. The lexicon is distributed under the proviso that it must not be sold or be made part of any product that is sold, and that any additions made to it by the user follow the entry conventions. The reverser program (written in 'C') is also freely available.

Processing Stages

In this section a brief overview of the *kern* processing stages is given, and then more detail is provided for each of them. This section will therefore provide a snapshot of what the program currently does in order to produce the English output; it is a “warts and all” description that highlights both the existing functionality and the currently missing functionality (i.e. necessary future work).

First, the brief overview: The user starts *kern* at the LINUX/UNIX prompt and the program prompts for various pieces of information, such as the filename of the Cornish text to be translated. Having read the input Cornish text from its file into memory, and split it into separate sentences, *kern* first pre-processes the source Cornish text to replace single-sense multi-word Cornish phrases with their English equivalents. Word-for-word transfer of the remaining Cornish words then follows, using the lexicon described above. Several post-processing stages then occur, such as forming *-ing* participles, *this/that* detection, and disambiguation of certain closed-class Cornish words. It is planned that general word-sense disambiguation (WSD) will then occur in order to resolve any remaining multi-sense English words down to a single sense, but this has not yet been implemented. Finally, structural rearrangements will be performed.

Below is given an example short Cornish text (an article from the *Nowodhow Kernow* website), together with *kern*'s output translation of it. (Note that input sentence numbering starts at nought.)

```
KERN VERSION 49

KERN SHORT OUTPUT FILE FOR INPUT FILE 'scylla.kern'
SENTENCE STRUCTURE OF INPUT TEXT:-

   0 Dy Sadorn 27/3/04 : Wosa gortos termyn hir rag an gewer ha chekkya bos
     pubonan salow , HMS Scylla veu sedhys y'n mor dohajydh Sadorn .
   1 Lemmyn yma hi a'y esedh war leur an mor may hallo sedhoryon neuvya a-dro
      dhedhi , ha may hallo puskes hag enyvales an mor kavoes trigva nowydh
     .
   2 An gorhel o an diwettha a'y far drehevys yn Porth Dewnans , hag yth esa
     kesstriv rag kowethasow neb a'n jeva hwans a wul devnydh anedhi .
   3 Wortiwedh , An Gwithti Mor Kenedhlek a brofyas dew kans mil peuns rag
     prena Scylla .
   4 An gwithti yn Aberplymm a vynn hy usya a-vel rann a'y ober .
   5 Heb mar , Scylla a vydh marthus rag tornyaseth yn Kernow Soth Est , hag
```

```
     yma porthow kepar ha Logh ow towlenna tenna an sedhoryon ha negysyow .

TRANSLATED SENTENCES:

Saturday 27/3/04 : After to await long time for the weather and to check
 abode/bush/to be everyone safe/healthy , HMS Scylla was sunk in the sea
 Saturday afternoon .
Now she/it is seated on the seabed so that may be able divers to swim
around her|it , and so that may be able fishes and sea creatures to find
 address new .
The ship was the last of his/her/its type built in Devonport , and was
competition for societies who/some/any he/it had desire if/of to do
things from her|it .
Finally , The National Maritime Museum suggested/offered two hundred
thousand pounds for to buy Scylla .
The museum in Plymouth wants to use her/it as/like part of his|her|its
work .
Without a doubt , Scylla will be marvel for tourism in South East
Cornwall , and there is|are ports just like Lake programming/planning to
 pull the divers and businesses .
```

Now, the details of the post-processing stage:

Starting *kern*

The user enters *kern* at the LINUX prompt and the program starts up, displaying header text on the screen. The text announces the name and version number of the running program and prompts for the name of the file holding the Cornish text, and a unique ID to give the output file; defaults are used if the user presses RETURN. Secondary questions ask the user whether the input text has one sentence per line format, or whether sentences potentially run over several lines. (This allows the sentence-chopper to be turned off in the former case.) The user is also asked whether he/she wants to see "near miss" words from the lexicon for any unknown Cornish words encountered (this is a very useful feature when translating a text – but it can be turned off; for example, if *kern* is being run on a file known to contain all new words i.e. if it is being run simply to build the lexicon from a Cornish word list.)

Error messages are produced for any condition that requires this e.g. if the input filename does not exist, or the lexicon cannot be found etc. All interaction with *kern* is textual; yes/no questions can be answered with a single y/n keystroke; longer responses have to be terminated with the RETURN key. As the program runs, progress and information messages are displayed on the screen. The user must be present to answer any questions *kern* may generate – *kern* is an open-loop learning system where a human user "closes the loop".

Kern is extremely robust and almost never crashes. If *kern* needs to stop, for example because some essential system file is missing, or perhaps if the input file is too big[4], then relevant and specific error messages are displayed and the program comes to a controlled

4 i.e. more than 500 lines, or more than 500 sentences, with a maximum of 1,000 characters in any sentence or 400 characters on any line.

and tidy stop. Error messages and questions to the user are in plain English, not computerese. This makes the program very user-friendly and very easy to troubleshoot. Running speed is of course machine dependent, but an estimate of total run time (not including any interactive dialogues) of five Cornish words per second would not be far wrong for a modern LINUX machine.

Sentence separation

If the user has indicated that sentences might wrap over more than one line, a function is called to split the text into its sentences, which are then stored in an array for translation one after the other. (The array is filled directly if the user indicated that the text was already arranged as one sentence per line.) Sentence-end detection is done in a fairly simplistic manner that looks for obvious markers (full stops, question marks, quotes etc.) but with some attempt to avoid premature sentence chopping e.g. due to full stops within abbreviations and numbers. This works very well, but very occasionally a sentence is cut off in its prime. (However, sentence ends are never missed, unless the input text used layout (without punctuation) to end the sentence, e.g. for titles.)

The sentence is the basic unit of translation within *kern*. Sentences are numbered and they are translated in turn to make up the output text. However, at all times *kern* has access to the whole text, which will be useful for later WSD. The descriptions which follow therefore apply to each sentence in turn.

The multiword-phrase pre-processor

The *kern* pre-processor is designed to cut down on WSD work by immediately replacing any unambiguous fixed Cornish phrases with their English equivalents. For example, *mis Gortheren* means “July” in Cornish, and could never mean anything else. It is essentially lexical. Since the main lexicon only contains single orthographic Cornish words on the LHS of the equals sign, it made sense to create a pre-processor to handle single-sense multiword phrases, with a separate phrase lexicon (see excerpt in Appendix C). However, as its use became established, it became apparent that there was no real problem with having some slashed elements within the phrase's meanings, as these would eventually be disambiguated by later processing. There are, however, no multiword phrases present where the *entire* phrase can have more than one meaning. (Such cases are handled by doing WSD on the individual words making them up, by the main processing.)

Does the phrase pre-processor represent memory-based translation (MBT)? Not in the usually-accepted fashion. In MBT systems the separate phrases can be merged together in the target language, and variables e.g. NPs inserted, and a concerted effort is made to extract, automatically, new phrases or word-strings from new texts encountered (aligned with their translations). This does not happen with the *kern* pre-processor. All phrases are fixed (no variable elements are possible) and either they match part of the input source language sentence (in which case they are used) or they do not occur in the input sentence. Phrases are tried in the order that they occur in the phrase lexicon. This means that, unless the phrase file is first ordered with the longest phrases first, phrases-within-phrases are not possible i.e. no phrase can be a subset of another unless the longer phrase is listed first.

Currently the phrase lexicon is ordered manually, but it would probably be a simple matter to order the file by decreasing line length (whilst preserving the header comments – see Appendix C). The phrase-set is manually collected, and has about 1,000 entries at present. Most Cornish phrases are two or three words long, but there is no practical limit to the number of words in each entry (as long as the whole entry X = Y is not more than 400 characters long).

After the phrase pre-processor has run, some of the Cornish words in the sentence may have been replaced by English words. So that future stages can avoid trying to translate the already-English words into English, they are marked with a preceding asterisk, as in this example:

> BEFORE PRE-PROCESSING:
> Dy Sadorn 27/3/04 : Wosa gortos termyn hir rag an gewer ha chekkya bos pubonan salow , HMS Scylla veu sedhys y'n mor dohajydh Sadorn .
>
> AFTER PRE-PROCESSING:
> *Saturday 27/3/04 : Wosa gortos *long *time rag an gewer ha chekkya bos pubonan salow , HMS Scylla veu *sunk *in *the *sea *Saturday *afternoon .

Note that punctuation characters are treated as words. Two phrases occur next to each other in the above (sunk in the sea, Saturday afternoon). The asterisks are removed later on.

Lexical Transfer

After pre-processing, each of the Cornish words in turn is looked up in the lexicon and the entire RHS meaning string is placed in the growing output sentence. Thus, continuing with our example above, after this word-for-word transfer, we would have:

> AFTER WORD TRANSFER:
> *Saturday 27/3/04 : After to_await *long *time for the weather and/while to_check/used_to_check(3,s)/might_check(3,s) abode/bush/to_be everyone safe/healthy , HMS Scylla was *sunk *in *the *sea *Saturday *afternoon

If any Cornish word is not found in the lexicon exactly as it appears in the sentence (apart from word-initial letter case, which can be either upper or lower) then the user is prompted to enter the missing word's meaning string. This will then be added to the lexicon, and also used in the current translation. If the user has requested “near miss” prompting at the start of the run, *kern* displays a list of the top 24 near misses in terms of spelling, arranged with the closest match first. The near-miss algorithm used is the Ratcliff-Obershelp algorithm (Computing, 1992) which returns a score between 0% (a completely different word) and 100% (exactly the same word). The near-miss level is currently set at 65%, but this is being reviewed as the lexicon size grows, and may

become word-length dependent. The algorithm is very good at bringing up words having different initial letters but which are otherwise the same (i.e. mutated forms), and it is also good at bringing back words with different endings (e.g. plurals of nouns, adjectivals etc).

Each near-miss entry is numbered. If one of the near misses is in fact the same Cornish word, but mutated differently, and/or if one of the near misses given has exactly the correct meaning string, then all the user has to do is enter the number of that near miss. Otherwise, the English sense string must be entered in the correct lexicon format. Once the lexicon has been taught the new word, it will use it ever after.

In addition to the near-miss prompting, *kern* suggests how any unknown word *might* be de-mutated (to aid in finding it in a printed dictionary). However, *kern* does not try to decide whether the unknown word really *is* a mutated form of a dictionary headword or not – it merely suggests what the de-mutation possibilities are if it were to be a mutated form. This feature is helpful for users who are not familiar with the full mutation table.

One other set of assistance messages is also provided. Built into *kern* is knowledge of Cornish prefixes and common word-endings associated with parts of speech etc. So, for example, if the unknown word ends 'ek', the user is told that the unknown word might be an adjective derived from the stem preceding 'ek'. Very often one of the near miss words is in fact that stem, or perhaps a mutated form of that stem. For example, if the unknown word is *vlydhenek* but one of the near misses is *blydhen* ("year"), the user is essentially told that *vlydhenek* might be a mutated form of the word *blydhenek*, and that the ending 'ek' probably means that it is an adjective. It is then obvious that *vlydhenek* means "yearly" or "annual". Very often this assistance obviates the need for the user to look up the new word in a dictionary.

Below is given a section of output (from a text about the restauranteur Rick Stein, from the *Nowodhow Kernow* website) illustrating a typical encounter with an unknown Cornish word, to illustrate the above. In this case, the unknown word *dhisplegya* turns out to be a mutated form of a known word. Comments have been inserted by the author in angle brackets **<thus>** (and some parts have been emboldened), but all the other text is produced as part of the long output file written by *kern*.

```
START OF TRANSLATED TEXT

SENTENCE 0
Dy Gwener 30/4/04 : Keginer a-vri der an bellwolok , Rick Stein , re
erviras kildenna dhiworth towlenn dhisplegya yn Tewynn Pleustri .
PRE-PROCESSOR: FOUND PHRASE 'der an bellwolok' AT POSITION 34 IN SENTENCE
PRE-PROCESSOR: FOUND PHRASE 'Dy Gwener' AT POSITION 0 IN SENTENCE
PRE-PROCESSOR: FOUND PHRASE 'Tewynn Pleustri' AT POSITION 114 IN SENTENCE

SENTENCE AFTER PHRASE PRE-PROCESSING:
*Friday 30/4/04 : Keginer a-vri *on *television , Rick Stein , re
erviras kildenna dhiworth towlenn dhisplegya yn *Newquay .

UNKNOWN KERNEWEK WORD DETECTED! FILLING SIMILAR-WORDS LIST - PLEASE WAIT...
```

```
THE UNKNOWN WORD WAS 'dhisplegya' - PLEASE GIVE ITS MEANING IN GERVA.TXT STANDARD FORM:
IN SENTENCE:
*Friday 30/4/04 : Keginer a-vri *on *television , Rick Stein , re
erviras kildenna dhiworth towlenn dhisplegya yn *Newquay .
TRANSLATED PART SO FAR IS:
*Friday 30/4/04 : Cook esteemed *on *television , Rick Stein ,
ones/by/too/(rel_ptl)/(vbl_ptl_re) decided(3,s)
to_withdraw/used_to_withdraw(3,s)/might_withdraw(3,s) from
programme/plan
SIMILAR WORDS TO dhisplegya IN GERVA.TXT ARE:
1 - 'displegya' which means 'to_develop/used_to_develop(3,s)' (94 percent sim.)
2 - 'displegyans' which means 'development/explanation' (85 percent sim.)
3 - 'displesya' which means 'to_displease/used_to_displease(3,s)' (84 percent sim.)
4 - 'displetya' which means 'to_display/used_to_display(3,s)' (84 percent sim.)
5 - 'displewyas' which means 'to_spread_out' (80 percent sim.)
6 - 'displegyansow' which means 'developments/explanations' (78 percent sim.)
7 - 'esplegya' which means 'to_evolve/used_to_evolve(3,s)' (77 percent sim.)
8 - 'displesyans' which means 'displeasure' (76 percent sim.)
9 - 'displetyans' which means 'display' (76 percent sim.)
10 - 'dastisplegyans' which means 'redevelopment' (75 percent sim.)
11 - 'plegya' which means 'to_fold/to_please/used_to_fold(3,s)/used_to_please(3,s)' (75
percent sim.)
12 - 'disklerya' which means 'to_declare/used_to_declare(3,s)' (73 percent sim.)
13 - 'dispresya' which means 'to_despise/used_to_despise(3,s)' (73 percent sim.)
14 - 'disesya' which means 'to_molest/used_to_molest(3,s)' (70 percent sim.)
15 - 'esplegyans' which means 'evolution' (70 percent sim.)
16 - 'plegyas' which means 'folded(3,s)/pleased(3,s)' (70 percent sim.)
17 - 'dilea' which means 'to_delete/used_to_delete(3,s)/might_delete(3,s)' (66 percent
sim.)
18 - 'dileshya' which means 'to_unleash/used_to_unleash(3,s)' (66 percent sim.)
19 - 'dilughya' which means 'to_demist/used_to_demist(3,s)' (66 percent sim.)
20 - 'diskleryans' which means 'declaration' (66 percent sim.)
21 - 'diskolya' which means 'to_de-carbonise/used_to_de-carbonise(3,s)' (66 percent sim.)
22 - 'disputya' which means 'to_argue/used_to_argue(3,s)' (66 percent sim.)
23 - 'disya' which means 'to_dice/used_to_dice(3,s)' (66 percent sim.)
24 - 'diveghya' which means 'to_unload/used_to_unload(3,s)' (66 percent sim.)
DE-MUTATION? The word 'dhisplegya' MAY be a mutated form, or it may not...
DE-MUTATION: This word starts with 'dh', so an unmutated form MIGHT start
             with a 'd'...
<At this point the user entered "1" to choose the first near miss's meaning string>
OK - will add 'dhisplegya = to_develop/used_to_develop(3,s)' to gerva.txt ...

SENTENCE BEFORE ANY POST-PROCESSING:
*Friday 30/4/04 : Cook esteemed *on *television , Rick Stein ,
ones/by/too/(rel_ptl)/(vbl_ptl_re) decided(3,s)
to_withdraw/used_to_withdraw(3,s)/might_withdraw(3,s) from
programme/plan to_develop/used_to_develop(3,s) in *Newquay .
<Processing continued here>
```

Post Processing Stage

At this point in the processing of the Cornish sentence, there is no Cornish left in the string being processed. The original Cornish sentence has been transformed, phrase-by-phrase and word-for-word, into an English 'sentence' which has the same word/phrase order as the original Cornish, but where each English 'word' may be a collection of English words separated by forward slashes (a "sense set"). For example:

> CORNISH SENTENCE:
> Lemmyn yma hi a'y esedh war leur an mor may hallo sedhoryon neuvya a-dro dhedhi , ha may hallo puskes hag enyvales an mor kavoes trigva nowydh.

ENGLISH BEFORE ANY POST-PROCESSING:
Now *she/it *is *seated on *the *seabed so_that may_be_able(3,s) divers to_swim/used_to_swim(3,s) *around *her|it , and/while so_that may_be_able(3,s) fishes and/while *sea *creatures to_find address new .

Thus all the processing done so far amounts to a word-for-word transliteration (plus maybe some full phrases) where all the possible senses of each Cornish word have been brought over into the English. It is the philosophy of the post-processing functions that they process only this English 'sentence'. The original Cornish is no longer used. Therefore, the lexicon entries needed to have contained all the necessary information in order for this to be possible. Let us consider now what sort of things need to be done in order to turn this 'sentence' into colloquial English.

Firstly, there is word-sense disambiguation, or, in our case, choosing which of the possible senses (as separated by slashes) is in fact the correct one in the current context. Many of the words in the English sentence will in fact be single-sense words (that don't need any WSD, therefore), but others may have two or more senses. In these cases, the one correct sense will need to be chosen, and all the others deleted. There are two sorts of WSD needed here. The first sort relates to common words such as *good/your(s)* where we must choose between two parts of speech, possibly including closed-class words. Here we have an adjective, "good", and the possessive adjective "your" (2nd person, s = singular). In Cornish, adjectives mostly follow their nouns, but the possessive adjectives precede them, as in English. This kind of situation can therefore be handled by examining the word order in the English 'sentence', and acting accordingly. In fact, many of the senses in this sort of situation can be immediately deleted using simple syntactical rules built into post-processing functions. (It is often fairly easy to see what the rules are, but tedious to code them.)

The second type of WSD is "true" WSD. Here, we may have a 'word' such as *case/hatred/war*. Clearly, we need to pick "case" for legal battles, "hatred" for a fewd between two people, and "war" for a conflict between two groups of people. All the senses are nouns, and we need to use some general WSD function to do the disambiguation. This function has not yet been written. It is envisaged that a thesaurus-based approach will be used, matching the possible senses to the context of the word in the sentence, and indeed the sentence in the whole text.

Apart from picking the correct word senses, we shall also need to perform structural transfer actions, to allow for the different grammatical structures in the two languages. The obvious one is moving adjectives from after their nouns to before them, e.g. *dog good* to *good dog*. But there are also "higher level" constructs to be handled. For example, "There is dog to me" needs to be processed to "I have a dog". Fortunately, it is the case that the English translation is usually understandable before such transformations, as often the Cornish word order or construction is not too strange to English ears. However, perfect grammatical transformation remains a desirable goal. At

the present time, only a few transforms have been done in *kern*. The reader is invited to examine the example translations given in this paper and to consider the transformations that might be performed, and how easy/difficult these might be.

Despite what has not yet been done, a certain number of fixed post-processing tasks have indeed been coded. These are listed in the table below. Each action has a name and an explanation of what it does. Many of these represent large coding efforts; certain problems/solutions are therefore mentioned in the Notes column of the table. Most of the effort is directed at deleting as many incorrect senses from the sense-sets ('words') as possible, hopefully down to the point where only one sense remains for each word. However, even partial WSD is useful as it cuts down the number of choices for "true" WSD to be performed later, and it makes the current output much more readable to the user, even without true WSD being performed. (In any case, the lexicon policy of only having one sense from a group of *closely*-related senses cuts down the true-WSD task.)

Post-processing function	***Actions***	***Notes***
Definite article	Removes verbal senses from the following word e.g. "the song/sings(3,s)" becomes "the song".	Removes many incorrect senses.
A-processing	Attempts to select the correct sense from: Oh!/goes(3,s)/if/of/(rel_ptl)/(vbl_ptl_a)	Complex, as there are six senses of the Cornish word *a*, as shown. Various syntactical clues are used to selectively delete senses from the string of six.
a'n processing	Selects the correct sense from of_the/(vbl_ptl_a)+himlit	Deletes the second sense if following word is not a verb requiring a particle.
y'n processing	Selects the correct sense from in_the/(vbl_ptl_y)+himlit	Ditto.
Indefinite article (Unn)	Examines context of "a_certain/one" because if followed by a VERB and preceded by "in" it means "VERBing" e.g. " came(3,s) in a_certain/one to_run" means "came running".	Where -ing form of a following verb has been made, it must be spellchecked. The "in" must also be deleted.

Post-processing function	***Actions***	***Notes***
Verbal particle 're'	Attempts to select the correct sense from: ones/by/too/(rel_ptl)/(vbl_ptl_re)	This function reduces the string to (rel_ptl)/(vbl_ptl_re) if the following word has all senses being a VERB that requires a preceding particle. (The string "has/have" also needs to be inserted; this is not done yet.)
This	Changes "the X so/that/here/is/are" to "this X"	Only works on one-word X at present. "These" will be done later, from "this X" where X is plural.
That	Changes "the X there" to "that X"	Being coded. Will only work on one-word X to start with. "Those" will be done later, from "that X" where X is plural.
Owth processing	Changes "(-ing_verb_follows) VERB" to "VERBing". e.g. "(-ing_verb_follows) to_drink" becomes "drinking".	The spelling of the English -ing form thus made also needs to be checked (e.g. *hitting* not *hiting*).
Orth processing	Changes "against/at/(-ing_verb_follows) his/its/(vbl_ptl_y) VERB" to "VERBing" when necessary.	-ing form made needs spell check.

Post-processing function	***Actions***	***Notes***
Ow processing	Changes "my/(-ing_verb_follows) VERB" to "VERBing" when necessary, amongst other things.	Complex. The Cornish *ow* is both an -ing marker and "my", but the construction "my VERBing" is also a desired result under certain circumstances. The words *y* and *hy* also need to be handled. Spellcheck also required.
And/while disambiguation	Chooses whether we need "and" or "while" from "and/while" e.g. "and she sitting" probably means "while she was sitting".	Cornish uses *ha* to mean both "and" and "while", in different grammatical constructs.
Active verb sense remover	If an active verb sense such as "eats(3,s)" is **not** preceded by a valid verbal particle such as "(vbl_ptl_a)", then the active verb sense is deleted, as it cannot be correct.	Removes many incorrect senses and hence aids full WSD which will take place later.
Remove verbal particles	Deletes (vbl_ptl_a), (vbl_ptl_re), (vbl_ptl_y) as these are not needed in English.	Done near the end of post-processing, when the information inherent in their presence is no longer needed.
Tidy-up	Removes stray asterisks, spaces, pipes and slashes etc	Housekeeping. Used at various stages.

Many other post-processing functions have been identified but not yet coded. These include "for to INFIN" processing (e.g. "for to see" becomes "to see"), adjective re-positioning as discussed earlier, adverb making (e.g. "in strong" becomes "strongly", although the phrase lexicon may be used for certain two-word adverbials, as they are essentially lexical), removal of repeated words (e.g. "to to" becomes "to"), and so on. In addition, all "higher order" transformations are yet to be performed e.g. "There|he|she| it_is John dancing" becomes "John is dancing".

It is clear that the post-processing stage is where the bulk of the work lies in a direct MT system. However, provided that someone is willing to investigate what is needed, and to create the code to do it, there appears to be no reason why good results should not be

possible. It must be stated, however, that it may well be that extra information (e.g. on PoS) may be needed in the lexicon in order to perform some of the necessary tasks. Moving adjectives in front of nouns, for example, will require PoS information. This might be provided by English noun and adjective lists, but this may not prove a viable solution.

Output

The output from *kern* is in the form of a short output file (kern.tout) containing the input and output sentences only, a medium-length output file (kern.mout) containing the sentences at various stages in the translation process, and a long (fully detailed) output file (kern.eng); and display onto the screen. The user is prompted for screen output choice at the end of the run, and also asked whether they want each output file printed. Screen output can be scrolled as it is piped to the UNIX *more* command.

The long output file contains full details of all the processing performed on each sentence, and is used to refine *kern*'s processing. The short output contains a copy of the input Cornish text, and the resulting English translation. An example of short output has already been given (the *Scylla* text). Part of the long output for this text is given in Appendix D.

Initial Results

The reader is invited to examine the short output for the *Scylla* text given earlier, and also another short output file given as Appendix E. Despite the presence of words not yet disambiguated down to a single sense (e.g. abode/bush/to_be), the meaning of the Cornish text can be deduced without too much difficulty. The *Scylla* text highlights various currently missing post-processing stages, including choice of anaphors based upon natural gender of the referent (his/her/its etc), verb number (is/are), some WSD (programming/planning) and some higher-level transformations ("wants its using" = "wants to use it"; "so that may be able X" = "so that X may be able"). NP and sentence subject detection is be needed, because subjects are often noun phrases, and knowledge of the subject's number and gender should help in choosing the correct anaphors and verb number. There is also one mistake: *Lake* should be the placename Looe, but one can perhaps forgive this, as in Cornish the town is indeed called "Lake" (Logh), for obvious reasons if one visits the place.

Kern now knows most of the words in any text new to it, (and in fact, *all* of the closed-class words) but there is usually still a handful of instances for any new text where the user has to supply the meaning(s) of an unknown word. This is hardly surprising at this early stage in lexicon building – with only about 9,000 Cornish headwords (including mutated forms, so that in dictionary terms there are actually many fewer than this). Every new text also throws up possibilities for new multi-word phrases, and building the phrase lexicon is a separate ongoing activity.

Despite the fact that it is still early days for the *kern* system, the program's ability to produce target language output that is understandable after a little deliberation is pleasing.

Future Work

New post-processing needs are being discovered continuously (some as new Cornish words enter the lexicon) and coded if they are feasible and would make a contribution to the quality of the output. An example of this is "with/by/hundred" (from the Cornish word *gans*, which as a dictionary headword means "with", but which might also be the headword *kans* mutated to *gans* ("hundred") – an example of this word occurs in the *Stein* text given in Appendix E.) In the vast majority of cases it will mean "with" or "by", but there are probably easily discovered syntactical rules which pick out the rarer "hundred" meaning.

One missing function is that of "a/an" insertion, clearly important in English, where it is used to signal an indefinite singular item (so the number of the entity involved will need to be discovered; plural indefinite entities need no marker, although "some" might be used). Another soon-to-be-coded function is the signalling in English of the perfective particle: "(rel_ptl)/(vbl_ptl_re) VERB-ed" needs to become "(rel_ptl) has/have VERB-ed", and then the relative particle has either to be deleted or replaced by who/which. Ultimately, more processing will then need to resolve the slashed cases within this, e.g. to "who has followed", "which have played", "which has come", "have eaten" etc. Clearly this depends upon the number and natural gender of the subject entity, and will require knowledge of irregular verb forms (e.g. "ate" becomes "eaten" in the perfect).

Phrase-level transformations, such as changing *There is (a) dog to me* to *I have a dog* and other sentence-scale word-order transformations, will be done later, at a time when WSD is advanced to the point at which very few multi-sense words are left in the output. For now, work concentrates on reducing each English word down to a single sense. Work also continues on building up the two lexicons, either by direct manual entry into the lexicon file(s), or by manual entry via *kern* (for single words) when a new text contains unknown words.

The *kern* system is a long-term, ongoing project. Along the way there may well be design U-turns and reworkings. For example, it may eventually prove necessary to indicate part of speech against English senses in the main lexicon. (But this will not be done unless it *has* to be done.) It may also be the case that the phrase lexicon comes to be more important than it currently is, pulling *kern* in the direction of memory-based translation (MBT). Along the way there will be opportunities for the author to explore well-known NLP problems, such as word-sense disambiguation (WSD). MT is in a sense a test bed for many, if not all, of the major NLP problem areas, such as anaphora resolution, SVO identification, NP-chunking, prepositional-phrase attachment etc. Some of these problems may turn out to have solved themselves – PP-attachment, for example, where the reader of

the English translation effectively solves the problem without realising that it is there! With the *kern* program already providing useful MAHT/HAMT functionality, the author looks forward to these higher-level challenges.

In the case of *kern*, the journey has only just started, and the author is sure that it will be an interesting trip. There will be further progress reports on *kern*.

References

BBC Cornish language website:
http://www.bbc.co.uk/cornwall/connected/stories/new_cornwall_language.shtml

Bowden, P. R. *Building a Lexicon for a Kernewek MT System* Procs. Workshop on Computational Resources for Minority Languages, part of TALN 2003 (Traitement Automatique des Langues Naturelles) (organised by The University of Nantes), Batz-sur-Mer, France (June 14, 2003) Also available on web:
http://www.sciences.univ-nantes.fr/irin/taln2003/articles/bowden.pdf

Bowden, P. R. *gerva.README* (A "read me" file describing the entry conventions used in the *kern* lexicon) – available from the author (paul.bowden@ntu.ac.uk). (2005)

Brown, W. *A Grammar of Modern Cornish* Kesva an Taves Kernewek ISBN 1-902917-00-6 (2001)

Brown, W. *Skeul An Yeth (Vols. 1, 2 and 3)* Kesva an Taves Kernewek (1996)

Cornwall County Council *Strategy for the Cornish Language (Stratejy rag an Tavas Kernewek)*, County Hall, Truro TR1 3AY. ISBN 1 903798 19 1. Also available at http://www.cornwall.gov.uk (2004)

George, K. *An Gerlyver Kres* Kesva an Taves Kernewek (1998)

Hutchins, W. J. and Somers, H. L. *An Introduction to Machine Translation* Academic Press (1992)

Just Cornish: http://www.justcornish.com

LER-BIML project: http://www.ling.lancs.ac.uk/biml/bimls3lang.htm

Nowodhow Kernow: http://geocities.com/cornishnews/

Warlinenn - website of Kowethas an Yeth Kernewek: http://www.cornish-language.org/

Ratcliff-Obershelp Algorithm Described in *Computing* newspaper, 20th August 1992, Notebook section, p.24 (1992)

Scannell, K. Website for *An Gramadóir*: http://borel.slu.edu/gramadoir/index.html
(An open source grammar checker; the Cornish port is being undertaken by Eduard Werner [edi.werner@gmx.de].)

Somers, H. *Machine Translation and Welsh: The Way Forward (Cyfieithu Peirianyddol a'r Gymraeg: Y Ffordd Ymlaen) A Report for the Welsh Language Board* [Centre for Computational Linguitics, UMIST, Manchester, England] (2004)

Tyack, J. *Developing A System to Parse Cornish (Kernewek Kemmyn) Text* Final Year Undergraduate Project Report, School of Computing and Informatics, Nottingham Trent University, Nottingham, England NG1 4BU (Held by the university library.) (2004)

Appendix A - A section of the *kern* lexicon

```
askallenn = thistle
askall = thistles
askel-groghen = bat
askell-dro = helicopter
askellek = winged
askell = wing/fin
askloesennow = splinters
askloesenn = splinter
askloesi = to_chip/to_splinter/used_to_chip(3,s)/used_to_splinter(3,s)
askloes = splinter/chip
askloetti = chip-shop
askloettiow = chip-shops
askorn = bone
askornek = bony
askorrans = production
askorras = product
askorrer = producer
askorr-lethek = dairy-produce
askorr = offspring/produce
askorroryon = producers
askra = bosom/pocket-fold
askrifa = to_ascribe/used_to_ascribe(3,s)
askus = excuse
askusya = to_excuse/used_to_excuse(3,s)
askusyow = excuses
aslammow = rebounds
aslamm = rebound
as = leaves(3,s)
asow = ribs
asper = harsh
aspia = to_observe/used_to_observe(3,s)/might_observe(3,s)
aspier = spy/observer
aspioryon = spies/observers
aspiyas = spy
aspiysi = spies
asrann = department
asrannow = departments
assa = How!/second
assaya = to_try/used_to_try(3,s)/might_try(3,s)
assay = attempt/essay/rehearsal
assays = attempts/essays/rehearsals
assentyans = assent
assentya = to_agree/used_to_agree(3,s)/might_agree(3,s)
ass = How!
assoylya = to_absolve/used_to_absolve(3,s)
astell-dhelinyans = drawing-board
astell-omborth = seesaw
astell = plank
astel-ober = strike
astel-omladh = ceasefire
astel = to_discontinue/discontinues(3,s)/discontinue!(2,s)
astelyer = striker
astelyoryon = strikers
astiveri = to_compensate/used_to_compensate(3,s)
astiveryans = compensation
astranj = strange/foreign
a's = (vbl_ptl_a)+her|it|them
asvens = Advent
```

```
asver = to_restore/restores(3,s)/restore!(2,s)
asvlas = aftertaste
aswa = gap
aswaow = gaps
aswek = gapped
aswels = regrown-pasture
aswonnans = acknowledgement
aswonnansow = acknowledgements
aswonnis = knew(3,s)
aswonn = to_know/knows(3,s)/know!(2,s)
aswonnvos = knowledge/recognition
aswonnys = known/you_used_to_know(2,s)
asyn = donkey/ass
asynes = donkeys/asses
asynigow = ass-foals
asynik = ass-foal
atalgyst = dustbin
atalgystyow = dustbins
atal = rubbish
Athelstan = Athelstan
a'th = (vbl_ptl_a)+you(s)/of_your(s)
atom = atom
atomek = atomic
atomow = atoms
attalow = returns
attal = repayment/recompense
attamya = to_broach/used_to_broach(3,s)
attendya = to_notice/to_pay_attention/used_to_notice(3,s)/used_to_pay_atten-
tion(3,s)
attent = attempt/experiment
attes = comfortable
atti = spite
attyli = to_repay
a-ugh = above
a-ughon = above_us
a-ughos = above_you(s)
a-ughov = above_me
a-ughowgh = above_you(p)
a-ughta = above_them
a-ughti = above_her|it
a-ughto = above_him|it
aval = apple
aval-bryansenn = Adam's-apple
aval-dor = potato
avalenn = apple-tree
avalennegi = orchards
avalennek = orchard
avalennow = apple-trees
aval-gwlanek = peach
aval-kerensa = tomato
avalow = apples
avalow-dor = potatoes
avalow-gwlanek = peaches
avalow-kerensa = tomatos
avalow-paradhis = grapefruits
avalow-sabenn = fir-cones
aval-paradhis = grapefruit
aval-sabenn = fir-cone
avalwydh = apple-trees
avalwydhenn = apple-tree
avalwydhennow = apple-trees
avanennow = raspberries
avanenn = raspberry
a-vann = above
avan = raspberry
```

a-var = early
a-varra = earlier
a-varr = early
a-vel = as/like
avel = as/like
avella = as_they|them
avelli = as_she|her|it
avello = as_he|him|it
avelon = as_we|us
avelos = as_you(s)
avelov = as_I|me
avelowgh = as_you(p)
aventurya = to_speculate/used_to_speculate(3,s)
a-ves = outside
avi = envy/liver
avis = advice/consideration
avisyans = notice
avisyansow = notices
avisya = to_observe/to_make_known/used_to_observe(3,s)/used_to_make_known(3,s)
avlan = unclean
avlavar = mute/speechless
avleythys = hardened/hard-man
avleythysyon = hard-men
avlymm = obtuse
a-vodh = voluntary/voluntarily
avodyas = left(3,s)
avodya = to_leave/used_to_leave(3,s)
avond = Begone!
avon = river
avonsyans = advancement
avonsya =
to_promote/to_progress/used_to_promote(3,s)/used_to_progress(3,s)/might_pro-
mote(3,s)/might_progress(3,s)
avonyow = rivers
a-vorow = tomorrow
a-vorrow = tomorrow
avoutrer = adulterer
avoutrers = adulterers
avoutres = adulteress
avoutresow = adulteresses
avoutri = adultery
avoutroryon = adulterers
avowa = to_acknowledge/used_to_acknowledge(3,s)/might_acknowledge(3,s)
avoweson = confession/admission
avoydadow = avoidable
avoydya = to_avoid/used_to_avoid(3,s)
a-vri = esteemed
a-wartha = above/on_top
awedh = watercourse
awedhyow = watercourses
awelek = very_windy
a-wel = in_the_sight_of
awel = wind/gale
awelyow = winds/gales
awenek = jawed/imaginative
aweni = to_inspire/used_to_inspire(3,s)
awen = jaw/inspiration
aweyla = to_preach/used_to_preach(3,s)
aweylek = evangelical

Appendix B - A section of the reversed lexicon

```
small = byghan/vyghan
small_cottage = dyji
small_cottages = dyjiow
smaller = byghanna
small_knoll = godolghynn
small_pit = dyppa
small_pits = dyppys
small_shield = bokler
small_shields = boklers
small_valley = golans
small_vallies = golansow
smell = blas
smith = gov
smithies = goveli
smiths = govyon
smithy = govel
smocking = hevisweyth
smoke!(2,s) = mog
smoked(3,s) = mogas/vogas
smokes(3,s) = meg/veg
smooth = gwastas
smoothing-iron = hornell
smoothing-irons = hornellow
snail = bulhorn
snails = bulhornes
sniffing = goverek
snow = ergh
snowflake = erghenn
snowflakes = erghennow
snowy = erghek
soap-opera = gwari-sebon
soap = sebon
so = bar/dell/mar/par
sober = divedhow
societies = gowethasow/kowethasow
society = gowethas/kowethas
soft = blin/bludh/isel/medhel
soft_ground = floukenn
soil = gweres/weres
solace = hebaska
sold(3,s) = gwerthas/werthas
sold = gwerthys
soldier = souder
soldiers = soudoryon
sole-fishes = gorleythennow
sole-fish = gorleythenn/gorleyth
sole = godhen
soles = godhnow
solidarity = unnveredh
solution = digolm
some = neb/nep
someone = nebonan
something = neppyth
sometimes = treweythyow
song = gan/kan
songs = kanow
son-in-law = deuv
son = mab
sons-in-law = deuvyon
```

```
Sons = Mebyon
soot = hudhygel
sooty = hudhyglek
sorceresses = hudoresow
sorceress = hudores
sore = goli
soreness = brewvann
sores = goliow
sorrow = ahwer/dughan/govijyon
sorrows = galarow
sorry = soweth
sort = eghenn/sort
so_that_he|it = ma'n
so_that = ma/may/mayth
soul = enev
souls = enevow
soundless = dison
sound-recording = sonskrif
sounds = trosyow
sound = tros
soup = kowl
south = deghow/dheghow/soth
south-east = deghow-est
southernwood = deghowles
South = South
southwards = a-dheghowbarth
sovereignty = myghternses
sow = banow/gwis
sower = gonador
sowers = gonadoryon
sows(3,s) = konis
sows = bynewi/gwisi
space = efander/spas
spaces = spasow
spacious = efan
spade = bal/pal
spades = balyow/palyow
Spain = Spayn
Spanish = Spaynek
spanner = alhwedh-know
spanners = alhwedhow-know
spark = elvenn/gwryghonenn
sparkler = elvennell
sparklers = elvennellow
sparks = elvennow/gwryghon
sparrow = golvan
sparrows = golvanes
speak!(2,s) = kows
speaker = arethor
speakers = arethoryon
speak_hoarsely!(2,s) = hos
speaks(3,s) = gews/kews
speaks_hoarsely(3,s) = hos
spear = gyw
spears = gywow
special = arbennik
specialism = arbennikter
specialist = arbenniger
specialists = arbennigoryon
speciality = arbennikter
speckled = brygh
spectacles = dewweder
```

Appendix C - A section of the Phrase lexicon

```
0000000 phrases.kern: a list of fixed, unambiguous multiword Cornish phrases.
0000001 Used by kern's phrase pre-processor.
0000002 ***** VERSION 015 *****
0000003 Don't put a space at the end of line! (Symptom: extra lone asterisks.)
0000004 All phrases (both Kernewek and English) must start with l.c. letter
0000005   unless they are always written as capitals. Code can make capitals!
0000006
0000012 Pad punctuation with spaces. (But no space at end of line.)
0000013 Be sparing with underscores in this file - use spaces between words,
0000014   but use slashes to separate word meanings (where you might have used
0000015   a vertical pipe in gerva.txt). There will be the odd exception to
0000016   this e.g. where you need X and Y_Z as separate senses: X/Y_Z
0000017
0000018 Paul Bowden (c) 2004
0000019
0000020 NEW APPROACH in VERSION 15 - DO NOT SORT THIS FILE!
0000021   Arrange manually - longer phrases must come first, where
0000022   there are shorter phrases which are their substrings, to
0000023   allow phrases-within-phrases to work! See e.g. "a vydh"
0000024   and "ev a vydh".
0000025
misyow a dhout = months of uncertainty
milyow a beunow = thousands of pounds
milyow a beuns = thousands of pounds
re ger o an pris = the price was too expensive
re ger yw an pris = the price is too expensive
re ger = too expensive
Yth esens i = They were
Yth esa = There/he/she/it was
an diwettha a'y far = the last of his/her/its type
a allav vy kavoes = can I have
a-ban eus kows dhodho = incidentally
a-barth a-woeles = down below
a-barth dhe = on behalf of
A-barth Dyw ! = For God's sake !
a-berth yn porth = in port
a-berth y'n = within the
a bub tu = in all directions
a dal = ought_to
a dalvia = ought to have
a-der bos = apart from being
a dhe = go/goes to the
a dhe'n = go/goes to
a dheu ha bos = becomes
a dheuth ha bos = became
a dho ha bos = used to become
a dhydh dhe dhydh = from day to day
a-dreus dhe = athwart
a dreylyas y woeles = overturned
a-dro dhe = about
a-dro dhedhi = around her|it
a-dro dhodho = around him|it
a-dryv dhe = behind
a dyb aga bos = believes them to be
a dyb agan bos = believes us to be
a dyb agas bos = believes you(p) to be
a dyb dha bos = believes you(s) to be
a dyb hy bos = believes her/it to be
a dyb ow bos = believes me to be
a dyb y vos = believes him/it to be
a'fedha = you(s) used to have
```

a'feu = you(s) had
a'fe = you(s) might have
a'fia = you(s) should have
a'fo = you(s) may have
a'fydh = you(s) will have
a fyll a = fail to
a fyllis a = failed to
aga honan = themselves
a'gan bedha = we used to have
a'gan beus = we have
a'gan beu = we had
a'gan be = we might have
a'gan bia = we should have
a'gan bo = we may_have/were_having
a'gan bydh = we will have
agan dew = the two of us
agan honan = ourselves
agan thri = the three of us
a'gas bedha = you(p) used to have
a'gas beus = you(p) have
a'gas beu = you(p) had
a'gas be = you(p) might have
a'gas bia = you(p) should have
a'gas bo = you(p) may_have/were_having
a'gas bydh = you(p) will have
agas honan = yourselves
a gynsa prys = for the first time
a hengov = traditionally
akont arghow = deposit account
akont kesres = current account
akont kreun = deposit account
akont poll = current account
akordya y golonn gans = agree with
alemma rag = henceforward
a leveris bos = said that (there) is/was/are/were
amari gweli = bedside cabinet
a'm bedha = I used to have
a'm be = I might have
a'm beu = I had
a'm beus = I have
a'm bia = I should have
a'm bo = I may_have/was_having

<more phrases follow>

Appendix D – Part of the long output file for the *Scylla* text

```
KERN VERSION 49

KERN LONG OUTPUT FILE FOR INPUT FILE 'scylla.kern'

OTHER IDENTIFIER ENTERED BY USER IS 'NONE'

<LINE STRUCTURE NOT PRINTED FOR SPACE/PRINTING REASONS>


NON-PRINTING CHARACTER FOUND IN INPUT TEXT - REPLACED BY A SPACE


SENTENCE STRUCTURE OF INPUT TEXT:-
SENT TEXT
---- ----
   0 Dy Sadorn 27/3/04 : Wosa gortos termyn hir rag an gewer ha chekkya bos
     pubonan salow , HMS Scylla veu sedhys y'n mor dohajydh Sadorn .
   1 Lemmyn yma hi a'y esedh war leur an mor may hallo sedhoryon neuvya a-dro
      dhedhi , ha may hallo puskes hag enyvales an mor kavoes trigva nowydh
     .
   2 An gorhel o an diwettha a'y far drehevys yn Porth Dewnans , hag yth esa
     kesstriv rag kowethasow neb a'n jeva hwans a wul devnydh anedhi .
   3 Wortiwedh , An Gwithti Mor Kenedhlek a brofyas dew kans mil peuns rag
     prena Scylla .
   4 An gwithti yn Aberplymm a vynn hy usya a-vel rann a'y ober .
   5 Heb mar , Scylla a vydh marthus rag tornyaseth yn Kernow Soth Est , hag
     yma porthow kepar ha Logh ow towlenna tenna an sedhoryon ha negysyow .

START OF TRANSLATED TEXT


SENTENCE 0
Dy Sadorn 27/3/04 : Wosa gortos termyn hir rag an gewer ha chekkya bos
pubonan salow , HMS Scylla veu sedhys y'n mor dohajydh Sadorn .
PRE-PROCESSOR: FOUND PHRASE 'Dy Sadorn' AT POSITION 0 IN SENTENCE
PRE-PROCESSOR: FOUND PHRASE 'dohajydh Sadorn' AT POSITION 117 IN SENTENCE
PRE-PROCESSOR: FOUND PHRASE 'termyn hir' AT POSITION 32 IN SENTENCE
PRE-PROCESSOR: FOUND PHRASE 'sedhys y'n mor' AT POSITION 103 IN SENTENCE

SENTENCE AFTER PHRASE PRE-PROCESSING:
*Saturday 27/3/04 : Wosa gortos *long *time rag an gewer ha chekkya bos
pubonan salow , HMS Scylla veu *sunk *in *the *sea *Saturday *afternoon
.

SENTENCE BEFORE ANY POST-PROCESSING:
*Saturday 27/3/04 : After to_await *long *time for the weather and/while
 to_check/used_to_check(3,s)/might_check(3,s) abode/bush/to_be everyone
safe/healthy , HMS Scylla was *sunk *in *the *sea *Saturday *afternoon
.

POST-PROCESSING STAGES:

DEF-ART PROCESSING: Found 'an' sense-string at position 52
DEF-ART PROCESSING: following string is 'weather'
SEPARATED SENSES:
   SENSE 0: 'weather'
DEF-ART PROCESSING: No (more) DEF-ART processing required in this sentence.
INDEF-ART PROCESSING: No (more) INDEF-ART processing required in this sentence.
A-PTL PROCESSING: No (more) A-PTL processing required in this sentence.
RE-PTL PROCESSING: No (more) RE-PTL processing required in this sentence.
THIS PROCESSING: No (more) THIS processing required in this sentence.
ING PROCESSING: No (more) OWTH processing required in this sentence.
ING PROCESSING: No (more) ORTH processing required in this sentence.
ING PROCESSING: No (more) OW processing required in this sentence.
HY-PROCESSING: No (more) HY processing required in this sentence.
```

```
Y-PTL PROCESSING: No (more) Y processing required in this sentence.
HA(G) PROCESSING: Found 'ha(g)' sense-string at position 63
HA(G) PROCESSING: following string is 'to_check/used_to_check(3,s)/might_check(3,s)'
SEPARATED SENSES:
   SENSE 0: 'to_check'
   SENSE 1: 'used_to_check(3,s)'
   SENSE 2: 'might_check(3,s)'
   ('dog' was NOT starred out.)
'to_check' is NOT a pronoun (or ***).
  Not every sense is a pronoun.
Choosing the 'and' sense of ha(g).
HA(G) PROCESSING: No (more) HA(G) processing required in this sentence.

POST PROCESSING: Sentence before INTERMEDIATE STRIPPING is:
*Saturday 27/3/04 : After to_await *long *time for the weather and/*****
 to_check/used_to_check(3,s)/might_check(3,s) abode/bush/to_be everyone
safe/healthy , HMS Scylla was *sunk *in *the *sea *Saturday *afternoon
.

INTERMEDIATE STRIPPING OF UNNECESSARY CHARACTERS...

POST PROCESSING: Sentence before PoS-BASED POST-PROCESSING is:
Saturday 27/3/04 : After to_await long time for the weather and
to_check/used_to_check(3,s)/might_check(3,s) abode/bush/to_be everyone
safe/healthy , HMS Scylla was sunk in the sea Saturday afternoon .

POST PROCESSING: PoS-BASED POST-PROCESSING is starting...

PoS-BASED POST-PROCESSING: Looking at word 'Saturday'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word '27/3/04'
IS_BRIT_DATE(): '27/3/04' is probably a date in Br or Am format.
  This is probably a date, so no further processing is required here.
PoS-BASED POST-PROCESSING: Looking at word ':'
  This is a punctuation mark, so no further processing is required here.
PoS-BASED POST-PROCESSING: Looking at word 'After'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'to_await'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'long'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'time'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'for'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'the'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'weather'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'and'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'to_check/used_to_check(3,s)/might_check(3,s)'
  This word has MORE THAN ONE SENSE, so further processing is needed...
SEPARATED SENSES:
   SENSE 0: 'to_check'
   SENSE 1: 'used_to_check(3,s)'
   SENSE 2: 'might_check(3,s)'
Word preceding 'to_check/used_to_check(3,s)/might_check(3,s)' is 'and'.
UNWANTED ACTIVE VERB SENSE REMOVER: Starting...
  CONTAINS_AVP(): 'and' does NOT contain an active-verb preceder.
  'to_check' is NOT an active verb requiring a valid preceder.
  IS_ACTIVE_VERB(): 'used_to_check(3,s)' is an active verb.
  'used_to_check(3,s)' is an active verb requiring a valid preceder...
  Examining context to see if this is possible:
  No - this is not allowed - the active verb sense will thereore be deleted...
    Done.
  IS_ACTIVE_VERB(): 'might_check(3,s)' is an active verb.
  'might_check(3,s)' is an active verb requiring a valid preceder...
  Examining context to see if this is possible:
  No - this is not allowed - the active verb sense will thereore be deleted...
    Done.
UNWANTED ACTIVE VERB SENSE REMOVER: Finished.

PoS-BASED POST-PROCESSING: Looking at word 'abode/bush/to_be'
```

```
  This word has MORE THAN ONE SENSE, so further processing is needed...
SEPARATED SENSES:
   SENSE 0: 'abode'
   SENSE 1: 'bush'
   SENSE 2: 'to_be'
Word preceding 'abode/bush/to_be' is 'to_check/*****************/***************'.
UNWANTED ACTIVE VERB SENSE REMOVER: Starting...
  CONTAINS_AVP(): 'to_check/*****************/***************' does NOT contain an act-
ive-verb preceder.
  'abode' is NOT an active verb requiring a valid preceder.
  'bush' is NOT an active verb requiring a valid preceder.
  'to_be' is NOT an active verb requiring a valid preceder.
UNWANTED ACTIVE VERB SENSE REMOVER: Finished.

PoS-BASED POST-PROCESSING: Looking at word 'everyone'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'safe/healthy'
  This word has MORE THAN ONE SENSE, so further processing is needed...
SEPARATED SENSES:
   SENSE 0: 'safe'
   SENSE 1: 'healthy'
Word preceding 'safe/healthy' is 'everyone'.
UNWANTED ACTIVE VERB SENSE REMOVER: Starting...
  CONTAINS_AVP(): 'everyone' does NOT contain an active-verb preceder.
  'safe' is NOT an active verb requiring a valid preceder.
  'healthy' is NOT an active verb requiring a valid preceder.
UNWANTED ACTIVE VERB SENSE REMOVER: Finished.

PoS-BASED POST-PROCESSING: Looking at word ','
  This is a punctuation mark, so no further processing is required here.
PoS-BASED POST-PROCESSING: Looking at word 'HMS'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'Scylla'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'was'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'sunk'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'in'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'the'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'sea'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'Saturday'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'afternoon'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word '.'
  This is a punctuation mark, so no further processing is required here.

REMOVING ANY REMAINING (vbl_ptl_y) AND (vbl_ptl_a) OCCURRENCES...

REMOVING ANY REMAINING (vbl_ptl_re) OCCURRENCES...

REMOVING ANY REMAINING (rel ptl) OCCURRENCES...

POST-PROCESSING: Sentence after post-processing stages is:
Saturday 27/3/04 : After to_await long time for the weather and
to_check/***************/*************** abode/bush/to_be everyone
safe/healthy , HMS Scylla was sunk in the sea Saturday afternoon .

REMOVING ANY (3,s) ETC REMAINING...

FINAL STRIPPING OF UNNECESSARY CHARACTERS...

SENTENCE AFTER ALL POST-PROCESSING:
Saturday 27/3/04 : After to await long time for the weather and to check
 abode/bush/to be everyone safe/healthy , HMS Scylla was sunk in the sea
 Saturday afternoon .
--------------------------------------------------------------------------


SENTENCE 1
```

```
Lemmyn yma hi a'y esedh war leur an mor may hallo sedhoryon neuvya a-dro
 dhedhi , ha may hallo puskes hag enyvales an mor kavoes trigva nowydh
.
PRE-PROCESSOR: FOUND PHRASE 'a-dro dhedhi' AT POSITION 67 IN SENTENCE
PRE-PROCESSOR: FOUND PHRASE 'a'y esedh' AT POSITION 14 IN SENTENCE
PRE-PROCESSOR: FOUND PHRASE 'leur an mor' AT POSITION 26 IN SENTENCE
PRE-PROCESSOR: FOUND PHRASE 'yma hi' AT POSITION 7 IN SENTENCE
PRE-PROCESSOR: FOUND PHRASE 'enyvales an mor' AT POSITION 113 IN SENTENCE

SENTENCE AFTER PHRASE PRE-PROCESSING:
Lemmyn *she/it *is *seated war *the *seabed may hallo sedhoryon neuvya
*around *her|it , ha may hallo puskes hag *sea *creatures kavoes trigva
nowydh .

SENTENCE BEFORE ANY POST-PROCESSING:
Now *she/it *is *seated on *the *seabed so_that may_be_able(3,s) divers
to_swim/used_to_swim(3,s) *around *her|it , and/while so_that
may_be_able(3,s) fishes and/while *sea *creatures to_find address new .

POST-PROCESSING STAGES:

DEF-ART PROCESSING: No (more) DEF-ART processing required in this sentence.
INDEF-ART PROCESSING: No (more) INDEF-ART processing required in this sentence.
A-PTL PROCESSING: No (more) A-PTL processing required in this sentence.
RE-PTL PROCESSING: No (more) RE-PTL processing required in this sentence.
THIS PROCESSING: No (more) THIS processing required in this sentence.
ING PROCESSING: No (more) OWTH processing required in this sentence.
ING PROCESSING: No (more) ORTH processing required in this sentence.
ING PROCESSING: No (more) OW processing required in this sentence.
HY-PROCESSING: No (more) HY processing required in this sentence.
Y-PTL PROCESSING: No (more) Y processing required in this sentence.
HA(G) PROCESSING: Found 'ha(g)' sense-string at position 116
HA(G) PROCESSING: following string is 'so_that'
SEPARATED SENSES:
   SENSE 0: 'so_that'
   ('dog' was NOT starred out.)
'so_that' is NOT a pronoun (or ***).
  Not every sense is a pronoun.
Choosing the 'and' sense of ha(g).
HA(G) PROCESSING: Found 'ha(g)' sense-string at position 158
HA(G) PROCESSING: following string is '*sea'
SEPARATED SENSES:
   SENSE 0: '*sea'
   ('dog' was NOT starred out.)
'*sea' is NOT a pronoun (or ***).
  Not every sense is a pronoun.
Choosing the 'and' sense of ha(g).
HA(G) PROCESSING: No (more) HA(G) processing required in this sentence.

POST PROCESSING: Sentence before INTERMEDIATE STRIPPING is:
Now *she/it *is *seated on *the *seabed so_that may_be_able(3,s) divers
to_swim/used_to_swim(3,s) *around *her|it , and/***** so_that
may_be_able(3,s) fishes and/***** *sea *creatures to_find address new .

INTERMEDIATE STRIPPING OF UNNECESSARY CHARACTERS...

POST PROCESSING: Sentence before PoS-BASED POST-PROCESSING is:
Now she/it is seated on the seabed so_that may_be_able(3,s) divers
to_swim/used_to_swim(3,s) around her|it , and so_that may_be_able(3,s)
fishes and sea creatures to_find address new .

POST PROCESSING: PoS-BASED POST-PROCESSING is starting...

PoS-BASED POST-PROCESSING: Looking at word 'Now'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'she/it'
  This word has MORE THAN ONE SENSE, so further processing is needed...
SEPARATED SENSES:
   SENSE 0: 'she'
   SENSE 1: 'it'
Word preceding 'she/it' is 'Now'.
UNWANTED ACTIVE VERB SENSE REMOVER: Starting...
  CONTAINS_AVP(): 'Now' does NOT contain an active-verb preceder.
  'she' is NOT an active verb requiring a valid preceder.
```

```
  'it' is NOT an active verb requiring a valid preceder.
UNWANTED ACTIVE VERB SENSE REMOVER: Finished.

PoS-BASED POST-PROCESSING: Looking at word 'is'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'seated'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'on'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'the'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'seabed'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'so_that'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'may_be_able(3,s)'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'divers'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'to_swim/used_to_swim(3,s)'
  This word has MORE THAN ONE SENSE, so further processing is needed...
SEPARATED SENSES:
   SENSE 0: 'to_swim'
   SENSE 1: 'used_to_swim(3,s)'
Word preceding 'to_swim/used_to_swim(3,s)' is 'divers'.
UNWANTED ACTIVE VERB SENSE REMOVER: Starting...
  CONTAINS_AVP(): 'divers' does NOT contain an active-verb preceder.
  'to_swim' is NOT an active verb requiring a valid preceder.
  IS_ACTIVE_VERB(): 'used_to_swim(3,s)' is an active verb.
  'used_to_swim(3,s)' is an active verb requiring a valid preceder...
  Examining context to see if this is possible:
  No - this is not allowed - the active verb sense will thereore be deleted...
    Done.
UNWANTED ACTIVE VERB SENSE REMOVER: Finished.

PoS-BASED POST-PROCESSING: Looking at word 'around'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'her|it'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word ','
  This is a punctuation mark, so no further processing is required here.
PoS-BASED POST-PROCESSING: Looking at word 'and'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'so_that'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'may_be_able(3,s)'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'fishes'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'and'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'sea'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'creatures'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'to_find'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'address'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word 'new'
  This word has ONLY A SINGLE SENSE, so no further processing is needed here.
PoS-BASED POST-PROCESSING: Looking at word '.'
  This is a punctuation mark, so no further processing is required here.

REMOVING ANY REMAINING (vbl_ptl_y) AND (vbl_ptl_a) OCCURRENCES...

REMOVING ANY REMAINING (vbl_ptl_re) OCCURRENCES...

REMOVING ANY REMAINING (rel ptl) OCCURRENCES...

POST-PROCESSING: Sentence after post-processing stages is:
Now she/it is seated on the seabed so_that may_be_able(3,s) divers
to_swim/***************** around her|it , and so_that may_be_able(3,s)
fishes and sea creatures to_find address new .
```

```
REMOVING ANY (3,s) ETC REMAINING...

FINAL STRIPPING OF UNNECESSARY CHARACTERS...

SENTENCE AFTER ALL POST-PROCESSING:
Now she/it is seated on the seabed so that may be able divers to swim
around her|it , and so that may be able fishes and sea creatures to find
 address new .
------------------------------------------------------------------------
```

<MORE OUTPUT DELETED HERE>

Appendix E – Short output for the *Stein* text

```
KERN VERSION 49

KERN SHORT OUTPUT FILE FOR INPUT FILE 'stein.kern'
SENTENCE STRUCTURE OF INPUT TEXT:-


   0 Dy Gwener 30/4/04 : Keginer a-vri der an bellwolok , Rick Stein , re
     erviras kildenna dhiworth towlenn dhisplegya yn Tewynn Pleustri .
   1 Yth esa govenek dhodho daswul an ostel Rocklands a-wartha Treth Tollkarn
      .
   2 Byttegyns , kost an projekt a ynkressyas dhiworth 1.5 milvil bys yn 6.5
     milvil .
   3 Drefenn na wra ev devnydh a skoedhyoryon arghansek kepar ha negysyow
     erell hag arghanntiow , re ger o an pris rag negys-teylu .
   4 Lemmyn , ev a vynn poeshe war y negysyow yn Lannwedhennek .
   5 Yma dhodho an Boesti Pysk , ostell , deli , ha lemmyn askloetti .
   6 Mes y ervirans re wrug dhe negysyow erell a Dewynn Pleustri pur serrys .
   7 Yth esens i hwansek a weles tus gans moy a arghans dhedha .
   8 Yma an dre owth assaya lesa hy les dres tus yowynk ha kluboryon , dhe
     deyluyow ha tus golusek .
   9 Rick Stein re leveris bos edrek warnodho wosa kildenna , hag ev re
     gollas milyow a beunow war an projekt ma .

TRANSLATED SENTENCES:

Friday 30/4/04 : Cook esteemed on television , Rick Stein , decided to
withdraw from programme/plan to develop in Newquay .
There/he/she/it was hope to him|it to restore the hotel Rocklands
above/on top Beach Tollkarn .
However , cost the project increased from 1.5 million all the way to 6.5
 million .
Because of he/it doesn't stuff/material if/of supporters
silvery/financial just like businesses others and banks , the price was
too expensive for business-family .
Now , he/it intends to bulk-up on his/its businesses in Padstow .
There is|are to him|it the Restaurant Fish , hotel , delicatessen , and
now chip-shop .
But/acorns/countryside his/its decision did to/for/at businesses others
if/of Newquay very angry .
They were desirous to see people with/by/hundred more if/of silver/money
 to them .
There is|are the town trying to spread her/its plant/use
over/beyond/besides/through the course of people young and clubbers ,
to/for/at families and people rich .
Rick Stein has said that (there) is/was/are/were regret/repentance on
him|it after to withdraw , while he/it/him lost thousands of pounds on
this project .
```