word. If the word is not in the list the word type with the most senses in WordNet for that word is chosen.
To enable the same (or a close as possible) set of synonyms to be returned for both the original word and the word that replaces it, as large a portion of the entire set of synonyms needs to be retrieved. This is done by retrieving the set of synonyms for all senses of the word that has been entered (whether it is to obfuscate or deobfuscate), using the word type found by the process described above. All of these senses are added to a list. The algorithm then goes through each word in this list in turn, and adds all senses for the current word type for each of those words, ignoring words that are already in the list. This then provides a list of a large portion of the words complete set of synonyms. The algorithm could repeat this process again, but with each level of abstraction from the original word the returned synonyms are less likely to be actually related to the word itself.

Data: word
Result: synset
synset = getsynonymsfromdictionary(word);
for $i \leftarrow 0$ **to** *synsetsize* **do**
| synset.addall(getSynonymsfromdictionary(synset[i]);
end
for $i \leftarrow 0$ **to** *synsetsize* **do**
| synset.setfrequency(getfrequency(synset[i-1], synset[i]);
end
synset.sort();
return synset;

Algorithm 1: Synonym Retrieval Pseudo-code

During this process, any senses which contain a single synonym are ignored, as well as any synonyms that are composed of two or more words, contain hyphens or are less than 3 characters in length. The words with spaces are ignored because it is impossible to know that they represent one synonym when deobfuscating. The words of less than 3 characters are ignored because in most cases they are either not in the dictionary, or they are in there representing an abbreviation for a name. For example, a synonym of "hello" is "hi", but "hi" also belongs in s synset with "Hawaii", which would obviously not be appropriate as a synonym as it has a completely different meaning to "hello".
Finally, this list is sorted according to the frequency of the bigram of that word and the word before it. The bigram frequencies are taken from the bigram frequency data from the ANC. This sorting ensures that the more likely synonyms to appear in the text are chosen above less likely ones.

Punctuation

Punctuation is not removed from the inputted text. A word such as "don't" will be searched as such. The reason for this is that it enables the algorithm to keep punctuation in the