Hobbs' Algorithm

First Author
Trey Feldman
tf912@nyu.edu

1 Introduction

Hobbs' Algorithm has long been used as a baseline system for pronomial anaphora resolution. Pronomial anaphora resolution refers to finding the antecedent for a given pronoun. For example, consider the following sentence:

John said the he loves himself.

In this sentence, there are two pronouns, he and him, both of which refer to their antecedent, John. In addition, pronouns may also refer to an antecedent in a previous sentence. Consider the following sentences:

Shelly got a new job. John said that she hates it.

In these sentences, she and it refer back to Shelly and job in the previous sentence, respectively. Lappin and Leass (1994) were able to implement a version of Hobbs' Algorithm that correctly resolved 82% of the pronominal coreferences attempted. The implementation described in this paper, however, faired much lower results. There are several reasons for this, which will be discussed later on.

2 Description of Algorithm

Hobbs' Algorithm takes as arguments a pronoun, the syntactically parsed tree that contains the pronoun, and syntactic parses of previous sentences. The algorithm, as described by Hobbs (1978), is as follows:

1. Begin at the noun phrase (NP) node immediately dominating the pronoun.

2. Go up the tree to the first NP or sentence (S) node encountered. Call this node X, and call the path used to reach it p.

3. Traverse all branches below node X to the left of path p in a left-to-right, breadth-first fashion. Propose as the antecedent any NP node that is encountered which has an NP or S node between it and X.

4. If node X is the highest S node in the sentence, traverse the surface parse trees of previous sentences in the text in order of recency, the most recent first; each tree is traversed in a left-to-right, breadth-first manner, and when an NP node is encountered, it is proposed as antecedent. If X is not the highest S node in the sentence, continue to step 5.

5. From node X, go up the tree to the first NP or S node encountered. Call this new node X, and call the path traversed to reach it p.

6. If X is an NP node and if the path p to X did not pass through the Nominal node that X immediately dominates, propose X as the antecedent.

7. Traverse all branches below node X to the left of path p in a left-to-right, breadth- first manner. Propose any NP node encountered as the antecedent.

8. If X is an S node, traverse all branches of node X to the right of path p in a left-to- right, breadth-first manner, but do not go below any NP or S node encountered. Propose any NP node encountered as the antecedent.

9. Go to Step 4

Put simply, the algorithm climbs the tree towards the root S node and does a breadth-first, left-to-right search on the subtrees rooted by S or NP nodes to the left of the path traversed. Any NP node encountered is proposed as the antecedent. If the antecedent is not found in the tree, a breadth-first search, left-to-right search is performed in a similar fashion on previous tree.

3 Proposals

For each NP node that is proposed as the antecedent, the algorithm must check whether the proposal agrees in gender and number. He, him, his, and himself all refer to males, and she her, hers, and herself all refer to females. In checking for gender agreement, the algorithm makes sure to reject an antecedent proposal of *Suzanne* for the pronoun *himself*. Additionally, the gender neutral pronouns it, its, and itself should not refer to *Suzanne* or any other name.

All pronouns mentioned so far refer to a singular person, place, or object, however, they, them, and theirs refer to a plural set of people, places, or objects. Thus, the algorithm rejects any proposals such as *Steve* as an antecedent for *their*.

The algorithm also considers several other preferences for anaphora resolutions through the order in which it proposes noun phrases. First, the algorithm considers constraints put forth by binding theory. Binding theory considers when and where in a sentence a pronoun can refer to it's antecedent. For example:

Angela bought a cake for her.

In this sentence, her can not refer to Angela because her is a non-reflexive pronoun and Jim is the most immediate subject in the sentence.

Secondly, the algorithm considers saliency. More recent noun phrases are checked by the algorithm first because they are likely more relevant to the pronoun trying to be resolved. Finally, grammatical roles are considered. If no antecedent is found in the immediate sentence for the pronoun *he*, then *he* likely refers to the subject of the previous sentence. The subject of the previous sentence is more commonly going to be found at the beginning of the previous sentence. Thus, the algorithm considers noun phrases from the previous sentence in a breadth-first, left-to-right manner.

4 Implementation

The algorithm was implemented in python and utilized several tools provided in the nltk package. Most importantly, parsed sentences from the Penn Treebank were used for implementing and testing the algorithm. These trees were labeled with part of speech tags and parsed according to the Penn Treebank grammar.

For checking number agreement, the part of speech tags, which indicate number, were used. For example, if an NP node was proposed that had a lone child labeled with an NNP (singular, proper noun) tag, the proposal would be rejected if the pronoun being resolved was *they*.

To check gender agreement, a corpus of male and female names was used. For example, a proposal of *Amanda* was rejected if Amanda was labeled as a female name in the names corpus, but the pronoun being resolved was *he*. Additionally, if the noun phrase contained Mr., Ms., or Mrs., the gender implied by the title was tested against the genders implied by the pronouns.

5 Testing

The algorithm was tested on the first 20 files found in the Penn Treebank corpus sample provided in the nltk package for Python. An answer key was handmade by the author for each pronoun found in the subset of the corpus. This allowed the program to compare the proposed antecedent of the algorithm to the true antecedent of the pronoun. The different files contained varying subject matter—from corporate legal matters to asbestos. The files also contained varying patterns of pronoun use. For example, some continually talked about one person. Thus, repeated uses of *he* or *she* often referred back to a name several sentences back. Other files switched around heavily between subjects. Some files were also heavier on using gendered pronouns and some contained more neutral pronouns. For example, certain pieces of text discussed various business entities which were referred to by *it* and *its* numerous times. These variations allowed for a more general solution to be tested versus specializing the algorithm for one sort of text.

6 Results

The results of this implementation were decent, but well off from the baseline achieved by Lappin and Leass. Overall, 74 of 116 pronouns were correctly paired to an appropriate antecedent for an accuracy of 63.79%. Between files, the accuracy had quite a bit of variance. In smaller files that contained less than 15 pronouns, accuracy ranged from about 33% to 100%. In larger files that contained more than 15 pronouns, accuracy ranged from a low of 53.5% to a high of 75%. Results for each file can be seen in Table 1

Male pronouns were resolved the most accurately at 75% accuracy (though there were only

two female pronouns to resolve, one of which was done accurately). Plural pronouns were close behind with 72.73% accuracy. Neutral pronouns did not fare so well, with only 59.68% accuracy. Only one reflexive pronoun was resolved, so the accuracy of the algorithm as it pertains to reflexive pronouns was not substantially tested.

File	Correct	Total	Percent Correct
wsj_0001	0	0	0%
wsj_0002	0	0	0%
wsj_0003	2	6	33.33%
wsj_0004	4	4	100%
wsj_0005	0	0	0%
wsj_0006	1	2	50%
wsj_0007	3	3	100%
wsj_0008	11	17	64.71%
wsj_0009	2	2	100%
wsj_0010	8	15	53.33%
wsj_0011	0	0	0%
wsj_0012	5	6	83.33%
wsj_0013	11	18	61.11%
wsj_0014	0	0	0%
wsj_0015	8	12	66.66%
wsj_0016	0	0	0%
wsj_0017	0	0	0%
wsj_0018	12	21	57.14%
wsj_0019	3	4	75%
wsj_0020	14	20	70%
Total	74	116	63.79%

Table 1: Results

7 Discussion

Due to the results being well below the baseline presented in Lappin and Leass, it is important to discuss improvements that could be made to get the algorithm up to that standard, as well as other improvements that may take it beyond that standard.

7.1 The Parse

One of the key components to the success of this algorithm is the parse of the trees and the algorithm's tailoring to the parse. Because pre-parsed trees were used without much knowledge of the parsing algorithm or grammar, it was hard to specialize the algorithm for the specific parse trees tested on. Much work could clearly be done to specify a grammar and parsing algorithm that is friendly to Hobbs' Algorithm.

7.2 Gender Checking

This implementation relied solely on a gendered list of names provided by nltk for gender checking. The issue with this was that, beyond names, a noun phrase such as *the girl* could also be an appropriate antecedent for *she*. Thus, it was hard to rule out any singular noun phrase for a male or female pronoun. This led to many results of noun phrases like *the car* being proposed and accepted for a pronoun such as *he*.

One possible solution this this issue is checking the hypernym of the head noun of the noun phrase. Thus, a noun like congresswoman, whose hypernym is female, could be judged as a female noun.

7.3 Prepositional Phrases

When a prepositional phrase at the beginning of sentence contains a pronoun, Hobbs' Algorithm has no way to resolve it correctly. For example:

Although he didn't want to, Jordan drove to the store.

In this sentence, because he is so early in the sentence, the algorithm will more than likely move to the previous sentence before it proposes Jordan as the antecedent. Thus, sentences like this always fail. To fix this, the algorithm would need to somehow check if the sentence begins with a prepositional phrase, and then suggest a noun phrase if it immediately follows a comma succeeding the prepositional phrase.

7.4 Reflexives

Because reflexives typically corefer to the subject in the most immediate clause, the proper subject is often not proposed due to the setup of the algorithm. Reflexive pronouns are often contained in the same noun phrase as the pronoun, and thus Hobbs' Algorithm does not consider them.

8 Conclusion

In general, the success of this algorithm clearly depends on the vetting of the proposed antecedents. With the algorithm alone and some simple checks, accuracy at around 63% can be achieved. To gain the extra 19% accuracy demonstrated by Lappin and Leass, a far more sophisticated algorithm for checking gender and agreement is necessary. Additionally, extra steps for checking reflexive pronouns as well as handling pronouns in prepositional phrases could really boost accuracy.

References

Jerry Hobbs 1978. Resolving pronoun references. *Lingua*, 44:311–338.

Shalom Lappin and Herbert J. Leass 1994. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561.