COMPARATIVE METHOD ALGORITHM

SOPHIA GILMAN

Advisors: Claire Bowern and Gaja Jarosz

Submitted to the faculty of the Department of Linguistics

in partial fulfillment of the requirements for the degree of

Bachelor of Arts

Yale University

1 May 2013

Abstract

COMPARATIVE METHOD ALGORITHM

Reconstructing proto-languages has been one of the goals of historical linguistics since the mid nineteenth century. The Comparative Method Algorithm (CMA) uses recent advances in computational linguistics to present a fully automated version of the classic comparative method used by linguists today. The program takes as input lists of words in related languages and a tree representing their relationship, and it uses this input to find cognate groups and reconstruct proto-words. For each reconstructed phoneme, the algorithm also provides a confidence estimate, which reflects the strength of the evidence for this reconstruction. The CMA provides a simpler, more flexible, and more transparent model than previous approaches, and it rivals those approaches in the quality of reconstructions produced.

Contents

Chapter 1

Introduction

In the last few years, several attempts have been made to automate parts of the phonological comparative method, a process historical linguists use to reconstruct sounds and whole words in the proto- language[1] of a given set of related languages.[2] This comparative method has been the "single most important tool" of the historical linguist since the 19th century and has seen "great success," especially in the realm of phonology (Trask 1996).

Yet, though the comparative method is effective, it is also exceedingly time-consuming. To take one example, Trask 1996 argues that for larger language families such as the Niger-Congo, "further generations of patient reconstruction" lie in store for linguists (p.218-9). Expediting such a process with a computational approach thus has the potential to save linguists around the world decades of work.

Even beyond speeding up reconstruction work, there is another issue a computational approach could help address. Currently, there is considerable debate in the literature as to what the reconstructions produced by the comparative methd represent; in what sense do reconstructed proto-words such as $*plHno$ actually represent a real language that someone somewhere once spoke (Trask 1996, Crowley 1992, Fox 1995)? To date, the issue is essentailly unresolved. Finally, lastly, it is also interesting, as a computation task, to determine whether a process as complex, though well defined, as the comparative method can be successfully automated.

[1] A proto-language is "the once spoken language from which daughter languages descend" (e.g. Latin is a proto-language for Spanish, Italian, etc.) (Campbell 1998, p.111)

[2] Some of the text in this section was drawn from Gilman 2012.

The few existing computational approaches to reconstruction have focused mainly on expediting the comparative method (Oakes 2000; Bouchard-Côté et al. 2013; List 2012b; etc.). In what follows, I present an algorithm for automating the comparative method (henceforth, the Comparative Method Algorithm or CMA) that aims to both help expedite the process of reconstruction and provide a new, more empirical approach to investigating the 'reality of reconstruction' question discussed above.

In what follows, I further discuss how the CMA addresses this 'reality of reconstructions' issue (Section 1.1), as well as present the heuristics that will be used in developing the CMA and comparing it with existing approaches (Section 1.2).

1.1 The Proposed Solution to the 'Reality of Reconstruction' Issue

The basic idea behind the CMA approach to the 'reality of reconstruction' question is to provide a sort of empirical measure of how 'real' a reconstruction is.[3][4] Currently, the debate on ranges over a variety of positions but is mostly theoretical. Formalists argue that such reconstructed forms are mere short-hand for describing what related languages have in common (Fox 1995; Pulgram 1959). Others believe they reflect something closer to true phonetic forms or that the only useful way to approach the reconstructive process is to act as if they do (Lass 1993). Most believe that reconstructed forms have some unspecified (and unspecifiable?) degree of phonetic reality (Trask 1996). None of these answers are particularly specific — the question is essentially unresolved. Campbell 1998, however, offers a more practical approach; he suggests that the realism of a reconstruction "depends on the material at hand to work with and the ability of the comparative linguist to figure out what happened" (p.147). The CMA will endeavor to provide a score that reflects precisely that — the quality of the evidence available and the confidence the algorithm has in the arguments advanced for the reconstruction it

[3] This notion is adapted from a similar but unimplemented idea mentioned in Steiner et al. 2011

[4] A significant part of the text in this section is drawn from Gilman 2012.

proposes. (Henceforth, this Degree Of Confidence Estimate is referred to as the DOCE.)

For example, if four related words in four different daughter-languages all start with a /t/, a linguist would be fairly confident in reconstructing a /t/ in the parent-language. Moreover, all but the most determined formalists would likely agree that the sound in the parent-language was a /t/. In such a case, the DOCE would be very high. On the other hand, if the first sound in each of the four daughter-languages was different (e.g. /h/, /s/, /z/, and /ʃ/), the linguist would be less confident and the formalists might argue that the reconstruction is less likely to represent a real sound. The more compelling the evidence (and thus the higher the DOCE), the more likely the linguistics community would be to accept the reconstruction as reflecting an actual reality.

Of course, in the classic (i.e. performed by human) comparative method, the decision making process is more complex than in the simplistic example above; it consists of a series of steps, at each of which the quality of the evidence (and thus the confidence of the linguist or algorithm) may vary[5] Much like the human historical linguist, the CMA walks through several steps, and, at each step, the CMA not only finds an answer (e.g. picks a proto-sound) but also provides a number representing the DOCE, reflecting the strength of the evidence for this particular decision. This last step is a way of quantifying an intuition that a human linguist does not usually explicitly express. Combining the DOCEs from all the steps, the final DOCE associated with a given reconstructed proto-phoneme will thus reflect the quality of the evidence for that reconstruction, and, by extension, be a measure of how 'real' a reconstruction is. Of course, calculating the DOCE for reconstructions is not a way to put an end to the 'reality of reconstruction' debate, but such an empirical approach may provide a new way of looking at the situation.

Thus, the defined goal of the Comparative Method Algorithm to be presented here is two-fold: to reconstruct proto-phonemes and proto-words based data sets from mod-

[5]The confidence of linguist can also be affected by a variety of other factors, such as the strength of the linguist's expectations or priors.

ern languages and to assign DOCEs to these reconstructions that would represent the strength of the evidence behind them.

1.2 Four Heuristics for CMA

The goal of this section is to present a set of heuristics that can serve as a guide in designing and evaluating the CMA algorithm to be presented in this paper, as well as compare it with previous approaches. In what follows, I describe the heuristics chosen here and give the reason for these choices.

simplicity This first heuristic commonly appears in linguistic theory as Occam's Razor (Abney 2011, p.14) and is a common "heuristic people use for comparing competing theories or models" (Eddington 2008, p. 12). A model that involves fewer components, steps, and/or variables but accounts for the same data as a more complex model is generally preferred to the complex model.

usability An algorithm that performs reconstruction is intended to be used by linguists to assist their own work. However, different linguists may have different approaches to the comparative method, and thus one potential goal of the CMA and any other similar algorithm is to be as flexible as possible, so as to be useful to the maximum number of linguists. For example, as Kessler 2005 points out, there different linguists hold a very wide variety of opinions on what sound changes are likely or natural; a reconstruction algorithm should thus ideally allow the linguist to choose whatever system they prefer to be used for reconstruction.

theoretical soundness The idea behind this last heuristic is that the CMA should evaluate the same sort of evidence the linguist uses in considering potential reconstructions. For example, if the linguist considers it important that sound changes be plausible, then the CMA should transparently consider whether a proposed proto-phoneme can plausibly change into its daughters as part of the reconstruction process. This last heuristic, however, is a result of two goals specific to the

CMA, and thus a different algorithm ignoring this heuristic is not necessarily flawed.

The first of these goals concerns the DOCEs. Recall from Section 1.1 that the DOCEs are assigned based on the evidence in the data set and the decisions made by the CMA, not by looking at the final reconstructed forms and evaluating them. This means it is important that the CMA actually consider the same types of evidence as the human linguist would in a transparent fashion. Thus, if a particular reconstruction is proposed, it should be clear to what extent this reconstruction fulfills the various requirements a linguist might set, so that DOCEs can be assigned.

The second reason behind having the CMA consider the same types of arguments as the human linguist is that such an approach would be an interesting alternative to the main current endeavor in the field of automatic reconstruction. This effort is a five-year project by four researchers (Alexandre Bouchard-Côté, David Hall, Thomas L. Griffiths, and Dan Klein), which takes a statistical approach to the task of reconstruction (Bouchard-Côté et al. 2013), which, in that particular implementation, leads to a somewhat opaque model (discussed further in Chapter 2). For example, rather than using any theory-based model of what sound changes are likely, they derive a new sound change model from each data set, using a variety of statistical tools (Bouchard-Côté et al. 2013, p.1, p.2). It would be interesting to determine whether a model that draws more on the human linguists' understanding of the processes involved has the potential to perform as well as or better than this purely statistical approach, and having the CMA transparently consider the same types of evidence as the human linguist is a way of investigating this question.

Thus, for both the sake of having effective DOCEs and to investigate thiss second issue, **theoretical soundness** will be a heuristic in evaluating the CMA, though not in comparing it to other algorithms.

Together, these heuristics form a solid groundwork for developing the CMA and analyzing exisiting algorithms. In what follows, the heuristics will be used to compare previous work in the field (Chapter 2) and to form the CMA algorithm (Chapter 3).

Finally, as part of this section, I define **effectiveness**, the method that will be used to evaluate the results of the CMA as well as compare those results with those of other algorithms (Chapter 4). The general goal of **effectiveness** is to evaluate the qualit of the reconstructions output by an algorithm. However, as there is no means of checking the proto-words reconstructed by an algorithm against the original words in the proto-language (the proto-language is essentially never available), the approximation used here, as well as in all similar works, is to compare the output of the algorithm to results of a human performing the same task (Oakes 2000; Bouchard-Côté 2013; List 2012). Thus, in comparing two algorithms, the algorithm whose reconstructed proto-words are closest[6] to the human's reconstruction for the same proto-language will be considered to be the more effective one. For the CMA itself, **effectiveness** also has another meaning, which concerns the DOCEs introduced in Section 1.1 above. This sub-heuristic, notably, cannot be used to compare the CMA to other algorithms, as there are currently no other works in the computational historical linguistics literature that make use of an idea similar to that of DOCEs.[7]. The DOCEs are meant to reflect the actual strength of the evidence for proposing the reconstructions returned by the CMA, and, thus, by extension, there should also be a correlation between the 'accuracy' of the reconstructions and their DOCEs. Thus, the DOCEs meet the **effectiveness** heuristic if (a) there is a correlation between the conclusiveness of evidence to the eye of a human linguist and the DOCE of the reconstructions for that data set and (b) there is a strong correlation between the quality of the reconstructions and the scores assigned

[6]For a more precise definition of the closeness, please see Section 4.2.2.

[7]Bouchard-Côté et al. 2013 mention that their algorith gives a posterior probability distribution for the reconstructions and sound changes their algorithm outputs. This score is not, however, used in any way in their work, so that the properties of this score are unknown and thus cannot be compared with that of the CMA's DOCEs

to them. This set of subhueristics will be used for the analisys of the results produced by the CMA in Chapter 4, as well as a heuristic for the comparison of algorithms in Chapter 2.

Chapter 2

Background Information

The classic comparative method can be described as a process, a series of steps, in which the linguist compares a group of daughter-languages to learn more about their proto-language. Here, the method is described as applied to phonological reconstruction, the attempt to reconstruct the sounds of the proto-language and combine those sounds into words. Notably, all the work discussed in this chapter and the CMA described in Chapter 3 focus solely on this phonological aspect of the method. Thus, for example, morphological processes are not modeled. The main reason for this is that, as Bouchard-Côté et al. 2013 point out, "phonological changes are generally more systematic than syntactic or morphological changes," and it is only this systematicity that makes automation possible (Bourchard-Cote et al. 2008: 1). While the next few years may see further developments in automatically reconstructing other types of change, the discussion here will focus on the purely phonological aspects of the comparative method.

While linguists disagree slightly on the best description of classic comparative method[1], the process can be broken down into three main steps. These steps are described below, along with the extent to which automatic approaches apply to them.

Identifying Hypothetically Related Languages to Compare The first step is identifying a set of languages that may have originated from the same proto-language. To do so, linguists look for languages that, essentially, share too

[1] See Trask 1996 for a description with seven steps, Campbell 1998 for one with six (phonology-related) steps, Durie and Ross 1996 for one with nine, and Fox 1995 for one with only three. All the authors, however, cover the same set of steps, but break them up in different places.

many characteristics, whether grammatical or phonological, for pure coincidence (Nichols 1995). While such coincidences are surprisingly common, languages that share, for example, certain complex grammatical features or have an already established set of related words can often be declared related with near certainty (Trask 1996; Nichols 1995).

This step relies on many different types of data and is here, as in other work, left to the linguist.

Furthermore, the CMA, as well as other work, relies on the user to provide the phylogenetic tree that further illustrates the relationship between the languages being compared, such as the partial tree in Figure 2.3 drawn from Lynch 2013.

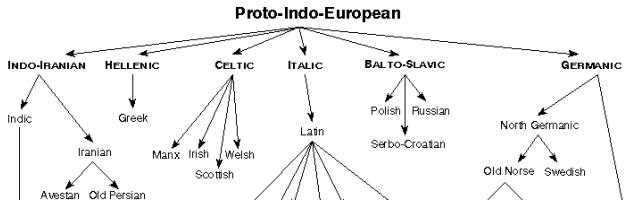

Figure 2.1: Part of a Phylogenetic Tree

The goal of such a tree "is to show how languages which belong to the same language family are related to one another" (Campbell 1998, p.187) The model suggests that "the ancestor language (a proto-language) developes dialects, which in time [...] become distinct languages (sister languages to one another, daughter languages of the proto-language), and how through continued linguistic change these daughter languages can diversify and slit up into daughters of their own (members of a subgroup of the family)" (Campbell 1998, p.188). (For a discussion of how phylogenetic trees are used in the reconstruction process, see Section 2.2.1).

In addition to such a tree, the user linguist is asked, for all existing algorithms, to provide data on each of the languages. This data, for all work discussed here, takes the shape of a set of words in each language. These words, notably, must for

Table 2.1: A Sample Input

meaning	Sinaugoro	Aroma	Hula
skin	opi	kopi	kopi
pigeon	pune	-	pune
die	mate	wareɣa	k^warea

the most part cover the same set of meanings, creating a database such as the one presented in Table 2.4. Both the human linguist and some algorithms (Steiner et al. 2011) can look outside a single meaning set for cognates, but the sets created by meaning serves as one of the criteria for establishing a cognate set.

Notably, as indicated by the dash, there might not be data for a some meaning in a subset of the languages. Furthemore, as the last row of Table 2.4 illustrates, the key is that words should have the same meaning, not necessarily that they should be phonetically similar.

Given these two inputs, the comparative method proceeds to the next stage, to which automatic methods can be applied.

Identifying Cognates and Correspondence Sets The goal of this step is two-fold. The first part of this goal is identifying groups of words within the input data which are cognate, meaning "related to their sister languages by [being] inherited by these sister languages from a common word [..] of the proto-language from which the sister languages descend" (Campbell 1998, p.112). Thus, for example, a linguist might identify /wareɣa/ and /k^warea/ to be cognates but put /mate/ in a separate group.

The second goal of this step is to identify sets of sounds that form correspondences, the sound equivalent of cognate sets. Formally, these are "sounds found in [...]

cognate sets, which correspond from one related language to the next because they descend from a common ancestral sound" (Campbell 1998, p.112). Table 2.2 shows the two words from our previous example aligned, so that each column forms such a correspondence set.

Table 2.2: Identifying Correspondences

Aroma	w	a	r	e	ɣ	a
Hula	k^w	a	r	e	[gap]	a

A sound correspondence seen in one word is furthermore "assumed to recur in various cognate sets," so a linguist might, for example, expect to see other languages where /w/ in Aroma corresponds to a /k^w/ in Hula (Campbell 1998, p.112). Finding such recurring or regular correspondences is the second goal of this step. Thus, one can consider any given column in Table 2.2 above an instance of a correspondence, and many different words can have instances of the same correspondence set (e.g. [/ɣ/, /Ø/] can appear as this type of column in many other words). In reality, the two goals (finding cognates and finding regular correspondences) are interweaved, and the process behind this step is described more fully in Section 2.1.1.

Notably, there is considerable literature on automating this step of the method. Given this, the CMA will not propose a new algorithm for this step but rather adopt the existing algorithm that best fits the heuristics described for the CMA. Comparing existing algorithms and choosing one to use for the CMA is the focus of Section 2.1

Reconstructing Proto-phonemes and Proto-words In this last step, the linguist uses the correspondences found in the first step to reconstruct proto-phonemes. Thus, for example, the proto-phonemes in the top row of Table 4.1 might be

reconstructed for the correspondences (shown in the columns of the bottom two rows as before) of /wareɣa/ and /kʷarea/.

Table 2.3: Identifying Correspondences

Proto	kʷ	a	r	e	ɣ	a
Aroma	w	a	r	e	ɣ	a
Hula	kʷ	a	r	e	[gap]	a

As with the 'Cognates and Correspondences' step, more details on the process involved in determining what the proto-phonemes should be are given later in the chapter, in Section 2.2.1.

These reconstructed proto-phonemes then fall together to create the proto-word for /die/: kʷareɣa (Campbell 1998, p.131). Notably, again, there might be much more complex morphological processes involved and the model where proto-words simply fall together from proto-sounds is overly simplistic. However, as the existing algorithms, including the CMA, only model purely phonological processes, this is the model used in the discussion in this chapter.

This final step of the comparative method has seen only two significant automation attempts: Oakes 2000 and Bouchard-Côté 2008 - 2013, both with a set of strengths and weaknesses (discussed in more depth in Sections 2.2.2 and 2.2.3 respectively). Given this, the main thrust of the CMA is designing a new algorithm for this final step of the method. Presenting such an algorithm will be the focus of Chapter 3.

As stated above, the rest of this chapter will focus on the second two steps of the comparative method and the attempts that have been made to automate them. In Section 2.1, more details will be given on how the human linguist executes the 'cognates and correspondences' step (Section 2.1.1), existing algorithms for automating

this process will be compared (Section 2.1.2), and the algorithm chosen for use in the CMA will be described in more detail (Section 2.1.3). In the second part (Section 2.2), the two existing algorithms for the 'reconstructing' step will be presented (Oakes in Section 2.2.2 and Bouchard-Côté et al. in Section 2.2.3), leaving the description of the CMA algorithm to Chapter 3.

2.1 Cognates and Correspondences

As stated earlier, there has been a significant amount of research dedicated to automating this step of the method. The goal of this section is simply to select the method that identifies both cognates and correspondences and best fits the four heuristics listed in the introduction.

To contextualize the discussion, an explanation of how the human linguist approaches this step is given in Section 2.1.1, before Section 2.1.2 introduces the state of the art in the field and Section 2.1.3 provides more detail on the algorithm chosen for the CMA.

2.1.1 The method of a human historical linguist

As mentioned in the introduction to this chaper, the input to this step is a list of words in the related languages, covering the same set of meanings, as in Table 2.4.

Table 2.4: A Sample Input

meaning	Sinaugoro	Aroma	Hula
skin	opi	kopi	kopi
pigeon	pune	-	pune
die	mate	wareɣa	k^{w}area

In that word list, the linguist searches for cognates and correspondence sets, using a somewhat recursive process (List 2012b). First, the linguist creates an initial list of potential cognate words, based initially mainly on the plausibility of the phonological pathways implied by the proposed word pairings (i.e. if /wareɣa/ and /k^{w}area/ are paired together, the linguist considers whether /w/ and /k^{w}/ could have come from the same proto-sound) (Fox 1995; Wilkins 1995). Sometimes, the linguist might also look at words with different meanings (rather than the same meaning set as above) if the meanings and phonemic forms of the words are sufficiently similar (Steiner et al. 2011). As Campbell 1998 points out, the linguist must also be very careful to avoid mistaking borrowings for cognates (words such as the English croissant that were borrowed from French, rather than evolving from the same proto-word as the same French word) (p.113). I note this, becuase, unfortunately, none of the cognate and correspondence identification algorithms currently available deal directly with the possibility of borrowings (List 2012b, Steiner et al. 2011; etc.). There is some other computational linguistic work that models language contact (e.g. Warnow et al. 2006), but these models are very complex and focus on inferring the phylogenetic tree (discussed below) rather than cognate and correspondence identification. Instead, the user linguist is recommended to use word lists from the basic vocabulary (low numbers, body parts, close kinship terms, etc.), which are considerably less likely to have such borrowings (Campbell 1998, p.112).

Using the initial list of cognates, the linguist creates a list of the regular (i.e. occurring in many words) sound correspondences among those cognates. When the list is complete, the linguist can edit or re-create the cognate list, now using the correspondence sets as the basis of evaluating how likely the words are to be cognate: if two words share what the linguist has identified as a regular correspondence (e.g. /w/ and k^{w} from above, if that pairs occurs in many words), then those two words are more likely to be cognate (List 2012b). Using *regular* correspondences in this way as a basis for establishing cognancy is better than using purely phonological or semantic similarity, as chance regular correspondences are much less likely than chance resem-

blances in form and meaning (Ringe 1992). Moreover, such misleading coincidences of form and meaning are surprisingly common, which makes the use of correspondences especially important (Trask 1996). As Trask 2000 points out, the insight that regular sound correspondences can be used to establish cognates is the great strength of the comparative method. After the linguist has created a new list of cognates based on the correspondences, the linguist can again update the correspondence set list based on the new cognates, and the process continues until no further changes to either the cognates or the correspondences are being made or until a sufficiently satisfactory result is reached (List 2012b). This iterative process is referred to hereafter as "learning from correspondences" in identifying cognates.

2.1.2 Comparing the Algorithms

The insights of this human method as well as the four heuristics developed earlier can now be applied to the task of comparing automations of this step, to pick one for use in the CMA. While there has been a vast amount of literature on the topic (and a full literature review is available in Delmesteri 2011), this section focuses on four recent proposals that capture the main advantages of past work: Kondrak and Hauer (2011), Delmesteri (2011), Steiner et al. (2011), and List 2012a and 2012b.

In the following discussion, the four algorithms mentioned above will be compared against our requirements and heuristics, and one of the algorithms will be selected to use for the CMA. Please refer to Table 2.5 throughout this discussion, where ‘V‘ indicates that an algorithm fulfills a given requirement and ‘X‘ indicates that it does not.

Table 2.5: Differences Between Four Algorithms

	Requirement	KandH	Del	Steiner	List
1	Finds Cognates	V	V	V	V
2	Finds Correspondences	X	V	V	V
3	Learns from Correspondences	X	X	V	V
4	Usability: Flexible Scoring Metric	X	X	X	V

The first requirement in selecting an algorithm is purely practical: it must identify correspondences, as well as find cognates. The next step of the comparative method, reconstructing proto-phonemes, relies crucially on correspondences, and thus, this step of the algorithm must provide them. All of the algorithms determine cognates (see row 1 of Table 2.5), but, as can be seen in row 2 of Table 2.5, there is one algorithm (Kondrak and Hauer's) that does not report correspondences. The method they use to identify cognates depends on a variety of string similarity measures, and their approach combines many of the different measures used in the literature (2011). While their approach makes the algorithm not usable for the CMA, such methods are a significant and thus noteworthy part of the computational tradition in identifying cognates. For the rest of this section, however, the discussion will center on the other three algorithms.

The next question then, captured in row three of Table 2.5, is whether the algorithm captures the crucial insight of the comparative method described in Section 2.1.1, learning from correspondences. Most of the earlier algorithms, however, do not do this, nor do they consider it: they fix on some sound similarity method for comparing words and establishing cognates, determine correspondences based on those cognates, and consider the process done (e.g. Kondrak an Hauer 2011; Delmestri 2011). These meth-

ods thus do not consider regular correspondences in establishing cognates at all and do not incorporate what Trask 1996 calls the chief advantage of the comparative method. An alternative, and better, method (used by List 2012a and Steiner et al. 2011) is to establish cognates at first based purely on sound similarity measures, determine regular correspondences, and then re-determine cognates incorporating the new data, just as the human linguist does. Notably, in fulfilling this requirement, these two algorithms also satisfy the **theoretic soundness** heuristic from Section 1.2, as these two algorithms consider the same evidence as the human linguist — the evidence provided by regular correspondences.

The next relevant heuristic is the **usability** of the model for different linguists, another one of the original four heuristics defined in Section 1.2. As discussed in Section 2.1.1, the human linguist originally formulates a preliminary list of cognates based on sound similarity, and this step requires the linguist to have some model of how the similarity of two sounds is to be evaluated. However, as Kessler 2005 points out, different linguists can have very different systems for doing so, and an algorithm that satisfies the **usability** heuristic should allow the linguist to choose their own system for determining such similarity. Three of the four algorithms being compared, as noted in Row 4 of Table 2.5, do not allow the user such a choice. Only List's algorithm offers the user a method for defining their own schema for scoring the similarity of a pair of sounds (List 2012b). (List also provides several scoring metrics for users to choose from, if they prefer not to define their own.[2] The other methods offer the user no options at all: Kondrak and Hauer 2011 neglect phonological similarity measures all together, preferring string similarity measures such as shared ngrams[3]; Delmestri 2011 uses a pre-set and very course sound similarity scoring metric (see p.72 for details); and Steiner et al.

[2]Even in List's algorithm, defining a new scoring metric is somewhat difficult and involves some slight modifications of the code. As a further note, the CMA uses one of the preset scoring metrics List offers — the SCA metric, which is based on an agglomerated set of phonological arguments and a sound class system (details in List2012a)

[3]Ngrams are substrings of the word of length n. Thus, /eɣ/ would be a bigram of /wareɣa/.

2011 initially distinguish only between vowels and consonants in their scoring metric, expecting the algorithm to learn all other distinctions. Thus, List's algorithm performs significantly better along the **usability** heuristic than any of the other options.

The last two heuristics from section 2.1.1 are **simplicity** and **effectiveness**. Unfortunately, neither of these are useful in this situation. The complexity of the algorithms does not vary sufficiently to discard any of them, and it is quite difficult to compare their effectiveness. None of the articles describing these methods compare their own performance to that of any of the other algorithms, either because others were released later or due to a lack of gold standard to use (as in the case of List and Steiner et al; List 2012, p.120). Thus, neither of these last two heuristics can be used.

However, even from the heuristics and arguments described above, it becomes clear that List's algorithm fits the heuristics and requirements of the CMA better than any of the other options, as the only one that fulfills every requirement. Thus, List's algorithm will be used for the 'cognates and correspondences' step of the CMA,[4] and the next section is dedicated to exploring the mechanics of this algorithm in slightly more detail.

2.1.3 Details of List's Algorithm

The previous section gave the broad outline of List's algorithm: it uses a sound similarity scoring scheme (metric) to create a preliminary list of cognates, finds the correspondences in this list, uses those correspondences to modify the scoring metric (i.e.learning from correspondences), and then makes another endeavor to find cognates. This broad outline is thus entirely in keeping with the method of the human linguist described in Section 2.1.1. In what follows, the more mechanical details of List's algorithm are described.[5]

First, the algorithm attempts to create a preliminary list of cognates. To do so, it makes a list of all word pairs with the same meaning. (List 2012a, p.121). For each

[4]Mattis List's algorithm is available publicly online and I have personal permission to use it for this project

[5]Much of the text in this subsection is drawn from Gilman 2012

pair, the algorithm uses dynamic programming[6] to find the way to match up the letters of the two words as in Table 2.6 (this is called alignment), so that the score of the alignment (discussed next) is as high as possible (List 2012a, p.121).

Table 2.6: Aligning 'king' and ''pig'

Word1	k	i	n	g
Word2	p	i	[gap]	g
Alignment score	0.5	1	0	1

The table shows the alignment of /king/ and /pig/ and, for each pair of aligned phonemes (e.g. /k/ and /p/) the sound similarity score they might be assigned in a scoring metric. In this table, it is assumed that higher similarity scores indicate more similarity, so that the scoring metric used here implies that /i/ and /i/ are very similar (in fact, identical), /k/ and /p/ are slightly less so, and /n/ and [gap] are not similar at all. The overall score is then calculated by combining these individual scores. In this case, the overall score of the alignment would thus be 0.5+1+0+1 = 2.5

If, on the other hand, the words were /king/ and /pit/, the alignment would depend on whether the scoring scheme said /g/ and /t/ or /n/ and /t/ were more similar. If /g/ and /t/, were more similar, the alignment on the left of Table 2.7 would be preferred to that on right of Table 2.7, since the score from the left part of the Table 2.7 alignment is better (1.8 rather than 1.6). 1.8 would thus also be the resulting 'sound similarity' score for that potential cognate pair.

Furthermore, in doing the alignment, the algorithm also identifies the potential sound correspondences — they are simply the columns in the tables above. The next

[6]Dynamic programming is essentially a method for breaking a given problem down into small steps. For a good explanation of how dynamic programming is used in application to alignment, see List 2012b.

Table 2.7: Aligning 'King' and ''Pig' in two Ways

Word1	k	i	n	g	Word1	k	i	n	g
Word2	p	i	[gap]	t	Word2	p	i	t	[gap]
Alignment score	0.5	1	0	0.3	Alignment score	0.5	1	0.1	0

goal of the algorithm is to find and learn from the regular sound correspondences.

The basic idea behind this step is that the algorithm attempts to compare the attested distribution of a given correspondence (pair of sounds in a column) against the expected distribution of that correspondence. The first step is calculating the attested distribution. For this step, the algorithm considers the pairs of words with the same meanings that have the highest similarity scores (threshold unspecified), and calculates the number of times every possible correspondence (pair of sounds) occurs in each language pair, as attested in this select set of word pairs (List 2012a, p.121). The next task is to find the expected frequency of each correspondence for each language pair. To do this, List's algorithm first shuffles each of the word lists repeatedly so that words with the same meaning are no longer necessarily on the same row (e.g. as in on the right side of Table 2.8) and then performs alignment on all pairs of words now on the same row.

Then, these alignments (without any selection for best alignments only) are used to count the necessary numbers: the frequency of each correspondence in each language. These numbers represent the attested distribution. Armed with these attested and expected distributions, the algorithm creates a new scoring scheme for each pair of languages. The sound similarity score of every two sounds in each such scoring scheme is calculated by comparing the attested distribution of those two sounds corresponding in those two languages against the expected distribution of the same sounds in the same

Table 2.8: A Sample Input

	Unshuffled					Shuffled	
Meaning	Sinaugoro	Aroma	Hula	Meaning	Sinaugoro	Aroma	Hula
skin	opi	kopi	kopi	skin	pune	kopi	k^warea
pigeon	pune	-	pune	pigeon	mate	wareɣa	pune
die	mate	wareɣa	k^warea	die	opi	-	kopi

languages. (For exact formula and discussion thereof, please see List 2012a, p.121). At this point, the new scoring metric is formulated, and thus the algorithm has learned from correspondences and is ready to recreate the set of cognates.

Having thus learned from correspondences, the algorithm now needs to identify a set of cognates again. First, it uses the new scoring metric to re-perform the alignments of all pairs of words that have the same meaning (i.e. no shuffling). (List 2012a, p.121). Now, there is an alignment score for each pair of words. The next task is to determine whether there are any groups of cognate bigger than a pair (e.g. /opi/, /kopi/, and /kopi/ are all cognate, but, in the /die/ group, only /wareɣa/ and /k^warea/ are). To do this, List uses a clustering algorithm called the flat cluster variant of the UPGMA,[7] which groups pairs of words into larger units based on all of the alignment scores of all pairs with the same meaning. These groups are considered to be the cognate sets. If a particular word is left without a group, it is in its own cognate set.

The remaining step, then, is to align these cognate word groups, so as to report correspondences to the user at the end of the process. This process differs from pairwise alignment mostly in purely computational ways, and at the end, List's algorithms has

[7]For a description of how this algorithm works, please see Sokal and Michener 1958. For the calculations in the CMA, the threshold was set to 0.5 and 0.55 for the Oceanic dataset (the data sets are described in Section 4.1.1)

a full list of cognate sets, each one presented as in Table 2.9 below, with the columns representing the correspondence sets.

Table 2.9: A Sample Cognate Set with Correspondences

Sinaugoro	-	o	p	i
Aroma	k	o	p	i
Hula	k	o	p	i

At this point, the correspondence sets and cognates have been found, and the next step — reconstruction of proto-sounds and proto-words — can begin.

2.2 Reconstructing Proto-Phonemes and Proto-Words

After the 'cognates and correspondences' step, the comparative method goes into its final phase: reconstructing proto-phonemes. As with cognates and correspondences, this section begins with an overview of the process used by the human historical linguist (section 2.2.1), followed by an overview of the work to date. In this case, only two significant attempts have been made to date to automate this step of the comparative method: Oakes 2000 (section 2.2.2) and a long term project by Alexandre Bouchard-Côté, David Hall, Thomas L. Griffiths and Dan Klein (2007; 2008; 2009; 2013) (section 2.2.3).

2.2.1 The method of the human historical linguist

At the reconstruction step of the process, the human linguist uses the correspondences found in the 'cognates and correspondences' step to reconstruct a proto-phoneme for each correspondence set.

One of the driving ideas behind this method is to reconstruct precisely one proto-

phoneme for each regular correspondence set and have a one-to-one relationship between corresondence sets and proto-phonemes (from here on out, this principle is referred to as **biuniqueness**) (Trask 2007: 263). The idea behind this principle is that sound change is regular (Trask 1996). Thus, if, for a certain word, a /k^{w}/ in the proto-language becomes a /w/ in one of the words in Hula, the same change should, generally, happen in every other word in the proto-language that contains /k^{w}/. It is possible that there might be some force (e.g. context, which is explained below) that forces k^{w}/ to become another sound (e.g. /k/) in Hula, but there would need to be convincing evidence. Thus, it is simply unlikely, that two different correspondence sets, which contain more than one phoneme, should have the same proto-phoneme, unless there was some reason for such a change. The most common such reason occurs when the correspondence sets occur in different contexts (i.e. the phonemes before and/or after those in the correspondence set are different).

Consider the following example, presented in Trask for reconstructing Western Romance (2007: 263). The linguist hs two correspondence sets: [/k/, /k/, /k/, /k/] and [/k/, /k/, /k/, /ʃ/]. However, the second correspondence is always followed by a correspondence set featuring a front vowel, whereas the first never is. Trask proposes the hypothesis /k/ > /ʃ/ | _ [front vowels] in the fourth language, and thus reconstructs /k/ in both cases. Furthermore, he argues that "on grounds of economy" it is in fact preferrable to reconstruct /k/ in both cases (Trask 2007: 263). In this case, as suggested above, there is a reason that /k/ > /k/ in some words and /k/ > /ʃ/ in others — the front vowel. However, baring such a situation, each phoneme in the proto-language is expected to correspond to precisely one correspondence set.

The human linguist also considers an array of factors in evaluating a potential proto-phoneme for a given correspondence set: *naturality* (a preference for reconstructing commonly occurring and/or theoretically likely changes), *majority* (if one sound predominates in the correspondence set, that sound is likely to be the proto-phoneme), a preference for reconstructing the minimal number of phonemes, and a preference for

not reconstructing any phonemes foreign to the language family. (Crowley 1992; Lass 1993; Trask 1995; Fox 1995)[8]. Finally, the linguist carefully notes how any data missing from the word lists affects the quality of his reconstructions (Trask 1995). To simulate the human, a reconstructing algorithm must, ideally, attempt to consider all (or at least most) of these factors as well.

Finally, as mentioned in the introduction to this chaper, the linguist considers all of these factors within context of the phylogenetic tree of the languages under consideration such as the partial tree inFfigure 2.3.

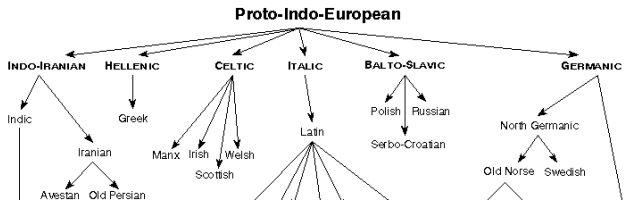

Figure 2.2: Part of a Phylogenetic Tree

For example, when reconstructing the proto-language of some group (e.g. Proto-Indo-European), the linguist (or group of linguists) would first reconstruct the various internal nodes of the phylogenetic tree (e.g. proto-Germanic, proto-Indo-Iranian, etc.) and use those to reconstruct Proto-Indo-European itself.

While both of the algorithms currently available rely on this tree model, as does the algorithm presented here, there are potential problems with using the model. Namely, the tree model rests on two false assumptions. The first is that "language splits are sudden" and the second is that there "there is no [...] contact among the related languages" after they first split (Campbell 1998, p.146). Both assumptions are widely considered to be wrong: languages diverge gradually and undergo changes from frequent contact with other languages within and without their language family (Trask 1996). However, as Fox points out, the first problem is not, in fact, a significant obstacle to use this model for reconstruction: the tree model representation may be simplified,

[8]This is by no means a complete list, but I believe it captures the most significant factors

but it accurately represents the linear relationships upon which reconstruction depends (Fox 1995, p.138-9). The latter issue, subsuming issues of borrowing and other forms of heavy language contact, is an admitted problem of relying on the tree model, though the human linguist has recourse to intuition and experience in avoiding such problems. However, as discussed in Section 2.1.1, it is an admitted problem of the existing computational models, including the CMA (Bouchard-Côté et al. 2013, p.5). The main way to ameliorate this issue is to use basic vocabulary as input to these algorithms, as basic vocabulary is less prone to borrowings. Thus, overall, the basic tree model is used here as a reasonable model of the comparative method, just as it has been treated as such by both Oakes and Bouchard-Côté.

2.2.2 Oakes' Reconstructive Model

In this and the following Section (2.2.3), the two main current attempts at implementing the 'reconstruction' step of the comparative method will be addressed. Throughout this discussion,please refer to Table 2.10 to compare the methods, strengths, and weaknesses of the human method against the Oakes and Bouchard-Côté et al.'s methods. Notably, the algorithm that performs better along each heuristic and/or sub-heuristic is highlighted in yellow. (**effectiveness** is included here even though it is more of an evaluatory metric than a heuristic, since it is still a useful way of comparing the two algorithms.)

The first significant attempt to automate this reconstructive aspect of the comparative method was proposed by Oakes in 2000. In what follows, this model will be discussed in detail with reference to the heuristics established in the Section 1.2 for comparison of different algorithms: **effectiveness** (rows 1-3 in Table 2.10), **usability** (rows 4-6 in Table 2.10, and **simplicity** (row 7 in Table 2.10). Recall that **theoretic soundness** was a heuristic to be used only in the construction of the CMA, not for evaluating other algorithms. Please consult the Oakes column Table 2.10 throughout this section for a summary of the characteristics of the algorithm. Notably, I here

Table 2.10: Evaluating Existing Methods On the Four Heuristics

	Heuristics	Oakes	Bouchard-Côté	Human
1	Effectiveness			
2	— Levenshtein Scores	2.33	1.86	0
3	— Scalability	4 Lang	637 Lang	Any with Diff.
4	Usability score			
5	— Naturality Options	Crowley's book	User Set	User Set
6	— Factor Weights	None	None	User Set
7	Simplicity	Very Simple	Millions of Parameters	Simple

focus, for both Oakes and, later, Bouchard-Côté et al., on the models used for their reconstructive algorithms, rather than the implementation, since the models capture the linguistic essence of the algorithms.

In Oakes' model, the basic goal of the algorithm was to model the comparative method as described in Crowley 1992 (Oakes 2000; p.233). Much like the human linguist, Oakes' algorithm looks at sets of correspondences (derived via an earlier portion of Oakes' algorithm) and endeavors to establish the correct proto-phoneme for each correspondence set (Oakes 2000; p.240). Unlike the human linguist, however, Oakes' algorithm assumes that the tree is flat, and the only task is to determine which candidate proto-phoneme best fits the root position of this flat tree, as shown in Figure 2.3.

This particular departure from standard theory is problematic, because this choice makes Oakes' model hard to use with a large set of languages. Consider the following

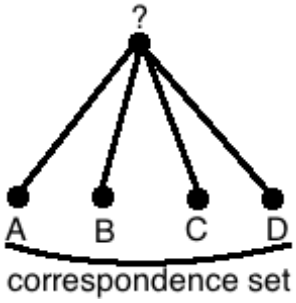

Figure 2.3: A Sample Correspondence Set and Tree

example. Suppose the algorithm has to reconstruct a proto-phoneme for this correspondence set: [/t/, /t/, /t/, /t/, /t/, /d/, /d/]. The algorithm might infer, using a flat tree, that the best candidate proto-phoneme is /t/, as it clearly performs the best on majority (i.e. most of the daughter phonemes are /t/). However, suppose the phylogenetic tree for the languages from which this correspondence stems is as in Figure 2.4.

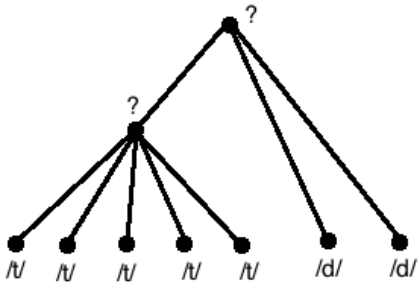

Figure 2.4: A Tree For the Correspondence

In that case, the proto-phoneme parent of all the /t/ phonemes should in fact be /t/. But, based on majority at least, the root proto-phoneme should be /d/, as two of the three of its immediate children are /d/. The same sort of analysis, though in a more complex way, applies with naturality. Oakes' model, however, cannot capture this type of complexity, and, as a result, cannot be particularly successfully applied to larger

trees, where similar situations are more common. For this reason, as Bouchard-Côté et al. 2013 argue, the Oakes' algorithm does not scale well, and, as shown in row 3 of Table 2.10, was applied only to 4 languages (Bouchard-Côté et al. 2013 Supplementary). Thus, this choice contributes to Oakes' algorithm performing worse on the heuristics of **effectiveness** for linguists (because it is less scalable), as well as effectiveness in general, becuase it misses an important level of complexity.

The next point of interest concerns the process used by Oakes' algorithm to reconstruct proto-phonemes, given this overall model. Oakes's algorithm essentially tries every phoneme in the phoneme inventory in the "question mark" places (the interior and root nodes) in Figure 2.3 and assigns each such phoneme a score. The remaining interesting question, then, is how such a score is calculated. Here, Oakes relies entirely on the model of sound change presented in the Crowley textbook, which, usefully, covers two of the factors used by the human linguist: *naturality* and *majority* (Section 2.2.1). More specifically, Crowley provides a list of sound changes considered natural (e.g. lenition), and Oakes uses that list to represent his *naturality* criterion (p.240).

A score is then assigned to each candidate proto-phoneme in the following fashion, illustrated (for two candidates) in Figure 2.5.

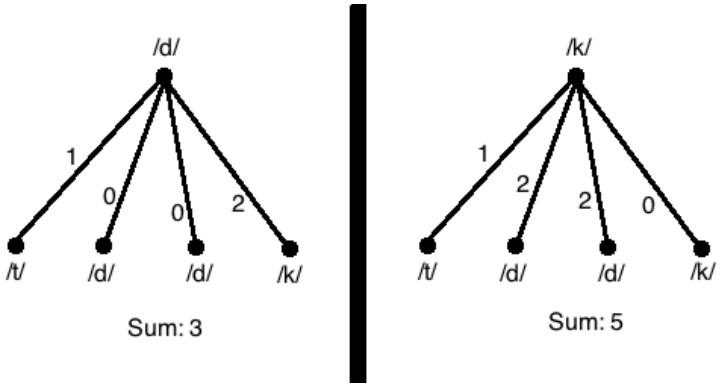

Figure 2.5: Evaluating Two Candidate Proto-Phonemes

First, a score is given to each branch of the tree as follows: the score is '0' is the candidate proto-phoneme and the leaf at the end of that branch are identical (e.g. /d/

> /d/), '1' if they are different but the change is one considered natural by Crowley (e.g. devoicing in /d/ > /t/), and '2' otherwise (e.g. /k/ > /d/) (Oakes 2000; p.240). Then, the overall score of that candidate proto-phoneme is simply the sum of the branch scores. The proto-phoneme with the lowest score (the one that thus requires the least costly changes) is chosen. Thus, in this case, /d/ would be a better proto-phoneme than /k/. This completes the description of the model.

The main advantage of this approach is it's fulfillment of the **simplicity** heuristic, as noted in row 7 of Table 2.10. This simplicity, however, comes at the cost of both of the other heuristics. First, the model does not satisfy the **usability** heuristic, as all of its elements are fully determined by Crowley's take on the comparative method, leaving no room for modifications to fit the theoretical beliefs of a particular user linguist. The two factors used in reconstruction that this algorithm accounts for, *naturality* and *majority* are lumped together, so that they cannot, for example, be weighted unequally, depending on the user's beliefs (row 6 of Table 2.10). Furthermore, *naturality* is defined precisely by Oakes' model, without any options for the user linguist, which is a significant detriment in terms of **usability**, as different users might have different phonological theories. Second, as Kessler 2005 points out and Oakes admits in the paper, the model is not particularly effective. As can be seen in row 1 of Table 2.10, the reconstructions produced by Oakes' model were on average farther from the human reconstructions than those of Bouchard-Côté.

Thus, while Oakes model does have the advantage of satisfying the **simplicity** heuristic, it cannot adequately satisfy either of the other applicable heuristics.

2.2.3 Bouchard-Côté et al.'s Reconstructive Model

The second, much more sophisticated, attempt at automating reconstruction was that of Bouchard-Côté et al. (henceforth BC). The project began in 2007 and is still continuing, with a recent paper in February 2013.

The basic model (i.e. the tree) used by BC differs in two important ways from the

model used by Oakes. First, unlike Oakes, this algorithm reconstructs words from set of cognate words, rather than reconstructing individual proto-phonemes from correspondence. In other words, the BC model holds that words evolve down a tree as shown in Figure 2.6. Notably, however, the BC algorithm still models only phonological sound changes of the same type considered by Oakes (Bouchard-Côté et al. 2013, p.2).

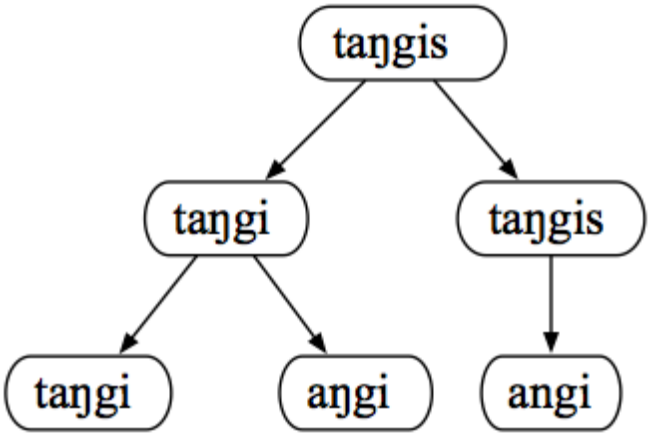

Figure 2.6: BC Model: Words Evolve Down a Tree

The main advantage of this view is increased sensitivity to context, which the BC model takes full advantage of, as it models sound changes based on context (2013; p.2). It is, however, a somewhat different approach from the class comparative method described in Section 2.2.1 in ignoring correspondence sets as a concept, whereas the classic comparative method relies crucially on that idea.

The second notable aspect of the BC model is that, unlike Oakes model, it allows trees with complex internal structure, such as the one in Figure 2.6. This, as discussed in Section 2.2.2, allows the model to be considerably more scalable scalable (BC ran it on 637 languages) and significantly reduces the average distances between the human and automatic reconstruction (Bouchard-Côté et al. 2013, p.1). Thus, the BC model performs better than the Oakes model on both aspects of the **effectiveness** heuristic (rows 2 and 3 of Table 2.10).

The final, and most complex, question concerns the sound change model that the BC algorithm uses to model the evolution of words. The main concept behind the BC algorithm is to avoid setting any variables (e.g. as Oakes sets the 'scores' of sound changes to '0', '1', and '2') and allow the learning mechanism of the algorithm to determine what the scores of each variable should be (Bouchard-Côté et al. 2013, p.2). The next question, then, is what the variables to be set are.

The model admits three types of parameters: operation, markedness, and faithfulness (Bouchard-Côté 2009, p.68). Operation variables are simply the probabilities of a particular type of 'operation' on a word as it evolves down a tree, where the 'operations' are insertion, deletion, substition (replacing a phoneme with another), and self-substitution (i.e. no change). In addition to these, particular changes (e.g. /d/ > /t/) can have probabilities, and these are considered faithfulness constraints. As a notable exception to the 'no pre-set variables' rule, a user can set particular faithfulness constraints in advance, thus defining the *naturality* factor. Finally, certain phonemes or combinations of phonemes (at most bigrams) can be unlikely to appear in a given language (i.e. at a given node in the tree), and that is considered a markedness constraint (Bouchard-Côté et al. 2009, p.68). Each of these variables can take on different values on each branch (operation) or at each node (markedness and faithfulness). Then, finally, this very large set of constraints is used to predict the probability of a given set of internal nodes and thus begin to evaluate the probability of a given root proto-word (Bouchard-Côté 2009, p.2)[9].

As Bouchard-Côté et al. point out, this model involves "literally millions of parameters to set" (Bouchard-Côté 2013, p.2), and this complexity clashes significantly with the **simplicity** heuristic, as noted in row 11 of Table 2.10. A model that could perform

[9]It is worth noting here that the BC algorithm can be used for considerably more than simply reconstruct proto-words, and the goal of the CMA is, for the scope of this project, **only** to complete reconstructions (Bouchard Cote 2013, p.4). However, if a simpler and better model can be found for this task, it may then be able to be expanded, while still remaining simpler, to encompass the other tasks.

as well as the BC model without invoking this number of variables would be a marked improvement.

On the other hand, the BC model fits both of the other heuristics reasonably well. It scales well and performs significantly better than the Oakes model (rows 2 and 3 of Table 2.10), thus satisfying the **effectiveness** heuristic. Unlike Oakes, the BC model allows users to define their own *naturality* variables, thus fulfilling that aspect of the usability heuristic (row 5 in Table 2.10). It is worth noting, however, that BC found that their model performs equally well without pre-set faithfulness variables and thus the later implementation of the models do not have pre-set constraints (including those that yield the results in Chapter 4) (Bouchard-Côté 2009, p.72). One potential problem is that, within this model, it is nontrivial to determine how various factors used by the human linguist are taken into account, though, many of them (e.g.*naturality*) likely are. However, as they are not explicitly distinguished, the user cannot, for example, assign them different weights, and in this sense the BC fails along that sub-heuristic of *usability* (row 6 in Table 2.10).

Overall, however, the BC model fits the heuristics surprisingly well. One of the main goals of the CMA model described in Chapter 3 will thus be to retain the advantages of the BC model, while improving on the **simplicity** and **usability** of the model.

Chapter 3

The Algorithm

The goal of this chapter is to present the CMA algorithm for the final, 'reconstruction' step of the comparative algorithm, as an alternative to the algorithms of Oakes and Bouchard-Côté et al. In addition, the algorithm is intended to calculate DOCEs, the confidence estimates defined in Section 1.1 that were meant to evaluate the quality of the evidence and considerations used by the algorithm.

Recall from Chapter 2, that the input to this final step of the comparative method is a list of cognates and correspondences within them, as found by List's algorithm (described in Section 2.1.3).

In what follows, I will present the sound change model (section 3.1) used by the CMA and then discuss the algorithm that uses this model to create both reconstructions and DOCES (section 3.2).[1]

3.1 The Model

Both Oakes' and Bouchard-Côté et al.'s models have their merits, and the CMA model (summarized in figure 3.1) attempts to capture all of these advantages, while minimizing the costs. Throughout the discussion, please consult Table 3.1, which extends the comparison made in Chapter 2 and Table 2.10 to cover the model used for the CMA itself. As in Table 2.10, the model that best satisfies each heuristic is highlighted in yellow.

[1] Again, some ideas and individual sentences in what follows are borrowed from Gilman 2012

Table 3.1: Oakes vs. BC vs. CMA

	Heuristics	Oakes	Bouchard-Côté	CMA
1	Effectiveness			
2	— Levenshtein Scores	2.33	1.86	2.15
3	— Scalability	4 Lang	637 Lang	120 Lang
4	Theoretical Soundness			
5	— Reconstructing From	Correspondences	Words	Correspondneces
6	— Factors	Two Main	Opaque	All Three Main
7	— Trees	Flat	Complex	Complex
8	Usability score			
9	— Naturality Options	Crowley's book	User Set	User Set
10	— Factor Weights	None	None	User Set
11	Simplicity	Very Simple	Millions of Parameters	Simple

In what follows, I present the two main aspects of the model: the type of phylogenetic tree assumed (Section 3.1.1) and the criteria used to model the way in which phonemes evolve down this tree (Section 3.1.2).

3.1.1 The Tree Used by the Model

The CMA involves two significant decision about the tree used that shape the algorithm: phonemes evolve down trees (with phoneme correspondence sets forming the leaves) and the tree can be complex.

First, the CMA model adopts the method of the human linguist and accepts a model where phonemes evolve down trees, and thus the goal is to reconstruct a proto-phoneme from each available correspondence set. The main issue with this, as mentioned in Section 2.2.3, is that not modeling words evolving down trees makes it more difficult to pay attention to context, as a human linguist would, which, as discussed in Section 2.2.1 is a necessary aspect of dealing with the *biuniqueness* factor. However, this disadvantage is neutralized in the CMA by the simple expedient of keeping track of the context for each correspondence set (more details in Section 3.2).

Second, like the BC model and the human linguist, the CMA model allows for complex trees (line 7 in Table 3.1). This, in turn, increases both sub-heuristics of **effectiveness**: the model scales reasonably well (row 2 in Table 3.1) limited mostly by memory requirements for List's algorithm, and performs between Oakes' and BC' model, though this will be discussed in much more detail in Chapter 4 (row 3 in Table 3.1). Furthermore, the CMA thus considers the problem from the same perspective as the human historical linguist, thus satisfying the **theoretic soundness** heuristic on both the choice of tree structure and the choice of character to evolve (see rows 6 and 8 in Table 3.1). This linguist-like perspective will allow the CMA to consider the same types of evidence as the historical linguist might, which, as discussed in Section 1.2, is a necessary pre-requisite to creating useful DOCEs.

These choices then yield a skeletal model something like the one portrayed in Fig-

ure 3.1.

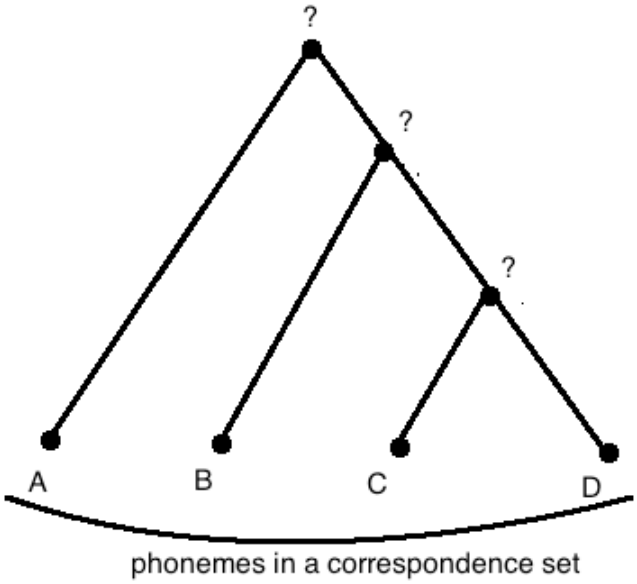

Figure 3.1: The CMA Sound Change Model

The leaves of this tree represent a set of phonemes that form a correspondence set (obtained, in the case of the CMA, from the alignments done by List's algorithm), and the root node is the proto-sound that the CMA must eventually reconstruct. The internal nodes represent other languages that are hypothesized to have served as intermediate stages in the evolution of the modern languages.

Given this model, the CMA should, just as the human linguist might, reconstruct values both for the nonterminal nodes and for the root node. Furthermore, all of the changes postulated must be proposed using similar types of evidence to that considered by the human linguist, as discussed in Section 1.1.

The next logical question is then what constitutes such evidence and how it can be evaluated. In other words, supposing that proto-phonemes were proposed for all of the internal nodes of the tree, how would such a tree be evaluated. The rest of this section will be dedicated to answering this question.

3.1.2 Criteria

Suppose the CMA has produced a set of candidates (X,Y,Z) to fill the internal nodes in Figure 3.1, as shown in Figure 3.2. In evaluating this tree, the model should evaluate each candidate proto-phoneme and the changes it creates (e.g. Z > Y or X > C) using the same types of evidence that a human linguist would use. The next goal, then, is to find some way of scoring each branch and internal nodes, using such evidence.

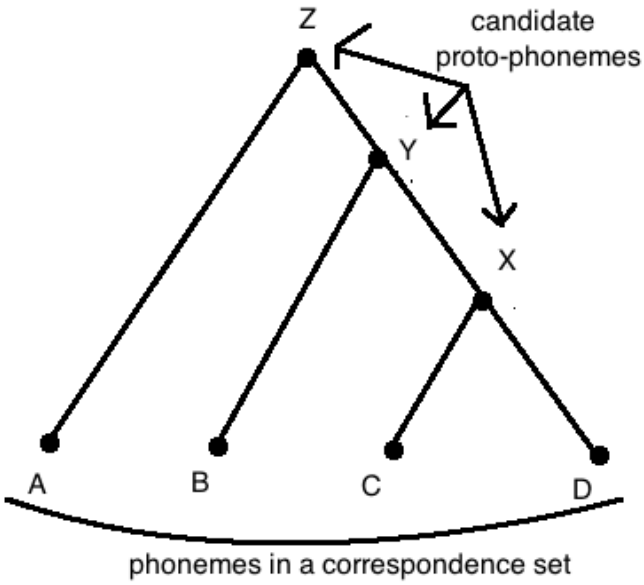

Figure 3.2: A Correspondence Set with Candidate Proto-Phonemes

With that aim, the scoring scheme discussed below combines a variety of factors commonly used in reconstruction (as discussed in Section 2.2.1) into a linear regression model and allows the user linguist to set the weights. Allowing the user to set the weights also helps the CMA to also satisfy all the aspects of the **usability** heuristic, as shown in row 6 of Table 3.1. In this, notably, the CMA model allows the user more flexibility than either the Oakes or the BC models.

The linear model created by these considerations is shown in equation (3.1). The weighted sum of these different factors is thus the DOCE of that particular candidate

proto-phoneme being assigned to that node.

$$DOCE = factor_1 * weight_1 + factor_2 * weight_2 + ... + factor_n * weight_n \quad (3.1)$$

After the DOCE of each candidate phoneme has been calculated, the overall DOCE of the set of candidate phonemes in the tree is just the product of their DOCEs' equation (3.2). Calculating the DOCEs in this way ensure that all proto-phonemes proposed by the CMA are well-supported by the evidence: if even one phoneme is ill-supported, the overall DOCE will plummet. On the other hand, one bad DOCE will not ruin the overall score of the model worked by averaging or adding the DOCEs of the nodes. Thus the overall score of the tree evaluates the reasonableness of that set of candidate phonemes as a hypothesis, and, by extension, the overall suitability of root proto-phoneme as the proto-phoneme of the modern languages.

$$DOCE_{tree} = DOCE_X * DOCE_Y * DOCE_Z \quad (3.2)$$

The remaining question, to which the rest of this section is dedicated, is establishing the factors from equation (3.1) that should go into evaluating a reconstruction. These factors are, as much as possible, the same ones as the linguist uses, as described in Section 2.2.1. The only criterion, from the list above, that is not used here is minimizing the number of phonemes reconstructed for a proto-language. Implementing this criterion is a potential project for future work, but the current version of the CMA focuses mostly on criteria that apply to a single correspondence set, and, as noted in row 6 of Table 3.1 and described below, it successfully implements all of the main criteria.

Below, I describe each of these factors, the theoretical justification for including them in the CMA model, and the way these factors are applied to data to, in each case, produce a number between 0 and 1 to serve as a 'factor' in equation (3.1).

Biuniqueness As discussed in the section on the method of the human in reconstructing proto-phonemes (Section 2.2.1), this criterion reflects a key insight involved

in the quest for regular correspondences: the goal of reconstruction is to propose a single proto-phoneme for each regular correspondence in the data, unless the two correspondences appear in complementary contexts (see Section 2.2.1 for an example).

The CMA model formalizes this intuition in the following way: there is a factor that suggests not reconstructing the same phoneme for two different correspondences **unless** the two correspondences appear in different contexts.

The next natural question, however, is how one defines context. In linguistics analyses, contexts of many different types appear. They can be on both sides, right only, or left only (e.g. # _, V_V, _#). They can be represented by a single phoneme (_p), by end- or beginning-of-word boundaries (#_), or by any complex sound description (_| [+voiced, +labial]), etc. In order to accomodate as many options as possible, the CMA model imposes costs of different strengths for whether a context is one-sided or two-sided. It also takes into account the word boundary marker #. The rest, in keeping with making the **usability** heuristic, is up to the user. The user can group symbols appearing in the input into any set of nonoverlapping classes (or no classes), and any set of phonemes belonging to that class will be lumped together for the *biuniqueness* calculation. Thus, for example, suppose context class B is [/b/ and /p/]. Then suppose the CMA has two different correspondences, one of which occurs in the _/b/ context and the other in the _/p/ context. They will be considered to have the 'same' right-side context for the purposes of this model, and thus attempting to reconstruct the same phoneme for both would warrant a penalty along the **biuniquness** criterion. Thus, again, users can choose any way of describing context that works best with other elements of their model.

The next question is how to quantify the *biuniqueness* score when evaluating a candidate proto-phoneme that has already been reconstucted for a different correspondence in a similar context. In the CMA model, this penalty varies

depending on the type of context involved. Suppose the candidate proto-phoneme is /b/ and a /b/ has already been reconstructed for a different correspondence set. The insight here is that the more closely the context of the candidate matches the context of the /b/ previous reconstructed, the heavier the penalty should be. Thus, if the context is matched on both sides (e.g. both correspondence sets occur in the #_i context, the penalty is heavier than if only the left or right side contexts match (e.g. #_ or _ɪ). The user can then choose how heavy the two penalties (one-sided context vs. both-sided context) will be. Then the score of the candidate proto-phoneme along this criterion is thus simply $1 - [penalty]$, so that it varies from 0 to 1 depending on the penalty assigned (where penalty can be '0' if no context conflict arises).

Naturality This criterion revolves around determining how likely one sound is to change into another, and, as shown in Section 2.2.1, it is one of the most important considerations in choosing a proto-phoneme. In the CMA model, this is equivalent to giving a score to a branch, of evaluating the probabililty of a (parent > daughter) change for each child of a given candidate proto-phoneme. This is the topic of much linguistic analysis, and the theories of what determines the plausibility of a sound change vary vastly. To give one example, Kessler 2005 explores a variety of sound comparison systems based on sub-phonemic features (e.g. place of articulation); he argues that the lists of such sub-phonemic elements "have been based on many different systems" and that a given system often represents a particular school of theoretical thought (2005: 249). The human linguist, to some degree, relies on a combination of experience and intuition, as well as formal theory, in making decisions (Trask 1996).

A computational model, however, must have some mechanical model of evaluation. Oakes solve this problem, as described above, by simply accepting the theory presented in Crowley's textbook (2000). Bouchard-Côté, on the other hand, give their model practically no a priori constraints on what changes are likely, although

the user has an option to preset such changes. The CMA models allows the user to fully determine what theory of *naturality* to use. This makes the CMA match the heuristic of **usability**, as the user is free to specify a model that matches their theoretical approach (see row 9 in Table 3.1).

Specifically, the user is asked to input a matrix that reflects their priors on likelihood of transition between every two symbols (on a scale of 0 to 1) that are used in the words the algorithm is asked to analyze (as in Table 3.2).

Table 3.2: A sample matrix with symbol set /A/, /B/, /C/

	A	B	C
A	0.7	0.2	0.1
B	0.3	0.4	0.1
C	0.1	0.1	0.8

The rows represent parent-phonemes, the columns — daughter-phonemes, and any given cell reflects the likelihood of the row parent-phoneme transitioning into the column daughter-phoneme. Thus, the user is entirely free to choose what, if any, theory they are interested in using to explore the data. For my own analysis, I have used two different approaches, and a discussion of their merits can be found in Section 4.1.2.

The numbers in the cells of the naturality matrix, thus, serve to provide a 'score' for each individual change involved in postulating a particular proto-phoneme. The overall score of a candidate is then simply the the sum of these scores divided by the total number of daughters. Thus, using the scores in the naturality matrix in Table 3.2, the model can evaluate candidate proto-phonemes along the *naturality* criterion given a correspondence set of (/A/,/B/). Thus, for example, the

score of /A/ as a candidate proto-phoneme is the average of the value the matrix gives for /A/ > /A/ and /A/ > /B/, where the row represents the proto-phoneme and the column represents the daughter phoneme. These same values (from the respective matrix cells) are shown on the branches of Figure 3.3. These calculations in Figure 3.3 thus show that /A/ is the proto-candidate with the highest *naturality* score.

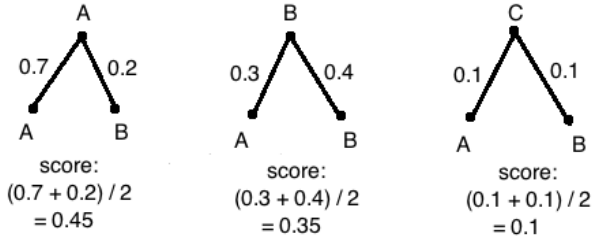

Figure 3.3: Using the Naturality Criterion

Majority This is the simplest of the criteria, but it is still considered to be in the top three most significant ones, as noted in Section 2.2.1. Essentially, a candidate proto-phoneme is better if it agrees exactly with more of its daughter phonemes (Trask 1996). Calculating the actual score for a proto-phoneme based on the majority criterion is very similar to the *naturality* calculation — but simpler. The score along each branch is a '1' if the candidate proto-phoneme is identical to that particular daughter-phoneme and '0' otherwise. At the end, the score for the majority criterion, is, again, the sum of these scores divided by the number of daughters.

Thus, if one is considering the correspondence set (/p/, /p/, /p/, /d/), the candidate proto-phoneme /p/ would score (3/4) on *majority*, /d/ would score (1/4), and any other candidate would score 0. While it is tempting to view *majority* as a subset of *naturality* (i.e. as the weight attached to unchanging phonemes), at least some of the literature suggests that the human linguist attaches a spe-

cial weight to daughter-sounds and candidate proto-sounds being identical (Trask 1996). Thus, in order to give the linguist maximum opportunity to adjust the model, the CMA creates a separate factor with its own weight for majority.

No foreign sounds This theory behind this criterion is that if a phoneme does not appear in any of the daughter languages, it probably does not appear in the proto-language either (Lass 1993). While there are known counter-examples (see Chapter 4 for details), this is still a useful general guideline. The score in this case is just '0' if the candidate is foreign and '1' otherwise.

No empty nodes reconstructed This criterion is purely practical and largely dependent on the data set being used. Very frequently, an insertion will occur in one of the daughter languages, so that a correspondence set of the form [/Ø/, /Ø/, /Ø/, /k/] appears. As can be seen from the points above, the majority criterion will bias the decision to reconstructiong a /Ø/. However, in some data sets, that is most often the wrong decision. Because this variable is data-set dependent, it is separated out from the *naturality* criterion, so that a user can use the same naturality matrix with different data sets and yet adjust this criterion. Thus, this criterion imposes a penalty on reconstructiong /Ø/ — it, too, is '0' if the candidate is /Ø/ and '1' otherwise.

Implicit: missing data This final criterion mentioned in Section **??** is implicit in the model itself, with no weight required from the user. As discussed in Section 2.1.3, the alignment part of the algorithm splits up sets of words with the same meaning into cognate sets, and correspondence sets are formed from these cognate sets. This means that any given cognate set, and thus any given correspondence set, might be missing words (and phonemes) for some of the languages.[2] The language tree the user provided, on the other hand, must contain all the languages.

[2]For a full discussion of how multiple cognate sets derived from a set of words with a single meaning are handled by the model, please see Chapter 4

Thus, fairly frequently, a situation such as the one in the right tree in figure 3.4 appears, where the '-' represents a missing data point (n.b. this is distinct from a known /Ø/). Note, however, that in both the *naturality* and *majority* score calculations, the sum of the scores is divided by the **total number of daughter phonemes**, rather than the sum of the scores. Thus, if a piece of data is missing, the proto-phoneme will receive a lower score, than if any phoneme was present. The difference created by a piece of missing data can be observed in figure 3.4.

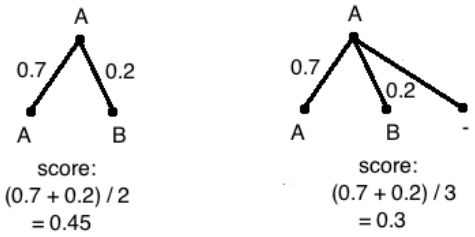

Figure 3.4: The Missing Data Criterion

This particular penalty, however, does not help distinguish between candidate proto-phonemes, as it creates a penalty independent of their scores. It does, however, ensure that missing data results in lower scores. This means that more missing data will result in lower DOCEs, which accurately reflects the notion that less evidence should lead to lower confidence estimates. At the same time, this penalty avoids penalizing certai proto-phonemes over others, which, given that the lack of evidence is not a good predictor, is also a positive effect.

Armed with these criteria and the method described above for combining them, the model can now evaluate and provide a DOCE for any given tree. Notably, this model successfully fulfills every sub-heuristic of both **theoretic soundness** and **usability**. Moreover, the CMA model also meets the **simplicity** heuristic (line 11 of Table 3.1). While the model is not as simple as that of Oakes, it is only slightly more complex, and the complications present are only such as are necessary to avoid two significant pitfalls

of Oakes' model: simple trees and somewhat opaque evaluation criteria. Furthermore, the CMA's use of complex trees, among other things, makes it an algorithm that reasonably fulfills the **scalability** sub-heuristic of **effectiveness** (line 3 in Table 3.1), which can also be considered part of **usability**, since an algorithm is more useful if it can be run on a larger number of languages. Thus, overall, the model seems to succeed in capturing the advantages of the previous algorithms, while avoiding their weaknesses. (The notable exception is the first sub-heuristic of **effectiveness**, and it will be discussed in more detail in Chapter 4.)

Given this model, the goal of the algorithm is to choose the set of proto-phonemes for each of the internal nodes (including the root node) that gives the overall tree the highest DOCE.

However, the space of hypotheses, especially for larger trees is too large to explore fully. Furthermore, the biunique relationship criterion means that trees are interdependent. In the section below, the algorithm used to efficiently explore this hypothesis space is described.

3.2 The Algorithm

The creation of an algorithm to implement the CMA model presents two basic challenges: exploring the too-large hypothesis space of a single tree and modeling the interdependencies between trees. Here, dynamic programming is used to address the former issue (Section 3.2.1) and a multi-run implementation of the algorithm deals with the latter problem (Section 3.2.2).

Relying on this second step of the solution (discussed in Section 3.2.2), the discussion in the next section takes as given that the relevant information on other trees is available and that the process of evaluating a single tree using the criteria described in the last section can be done in some polynomial time (call this function *evaluate tree*). Since the model section above showed how to assign a score to each node and combine those scores, this polynomial algorithm is trivial.

3.2.1 The Hypothesis Space and Dynamic Programming

The main problem with finding the best model is simply the size of the hypothesis space. The user supplies, via the naturality matrix described in section 3.1.2 a set of candidate proto-phonemes. Hypothetically, one would need to try each phoneme in this inventory at each internal node of the tree and try every possibly combination. This, however, supposing the size of the inventory to be a and the number of internal nodes to be n, would require the algorithm to evaluate a^n trees. An exponential algorithm of this type makes it impractical to run any large data.

The algorithm used here implements two time saving strategies. The first is fairly simple: rather than trying every possible phoneme in the inventory at every node, the algorithm only tries the the phonemes present in the leaf nodes and the x phonemes most likely to change into the leaf phonemes in the naturality matrix (the user can determine the value of x). The rationale behind this shortcut relies on the nature of the *majority* and *naturality* criteria from Section 3.1.2. If the user assigned significant weights to either of these criteria, then phonemes which are either in the leaf set (for *majority*) or likely to change into the leaf phonemes (for *naturality*) are the only ones with a reasonable chance of receiving high DOCE scores. Thus, if the user assigns high weights to *majority* and/or *naturality*, this shortcut is not likely to miss better trees. If, on the other hand, the user assigns very low weights to both *naturality* and *majority*, they can set x to be very high (e.g. the size of the inventory) and thus avoid the potential negative reprecussions of this shortcut. Since, as stated in Section 3.1.2, *naturality* and *majority* are often considered to be two of the most important criteria, most users should be able to take advantage of this shortcut.

The second shortcut used is algorithmic rather than theoretical and it relies on a simple, yet key, insight about the model:

insight: the DOCE of any internal node in the tree relies only on its immediate children (see section 3.1).

Of all the criteria, only *naturality* and *majority* involve children at all, so the insight

only needs to be proven vis a vis those criteria. It will in fact be illustrated only for *majority*, but the *naturality* situation is exactly analogous.

Suppose the model is evaluating the two trees in Figure 3.5.

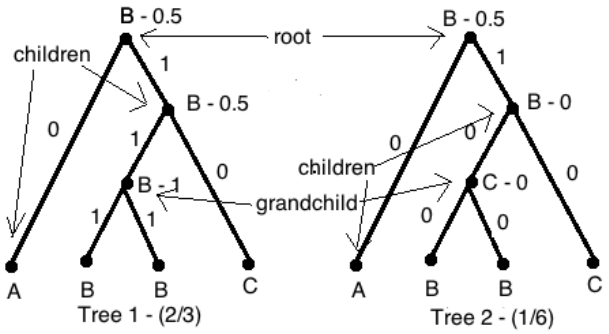

Figure 3.5: Evaluating Trees with Different GrandChildren

The branches are labeled with the *majority* score of the branch (‘1‘ for identical parent and child, ‘0‘ otherwise, as defined in 3.1.2), and the internal nodes are labeled with the majority score of that candidate phoneme at that internal node (the average of the branch scores). Consider first the calculation at the very top of the tree: for both trees, the root receives a 0.5 score for majority, as its children are identical in either case. Given the definition of *majority*, the root node is thus blind to the phonemes chosen for any nodes besides its immediate children. Suppose for a moment that the goal is maximizing the overall score of the tree without affecting the *majority* score of the root node. Given that the root node is blind to its grandchildren, it only makes sense to choose grandchildren that will maximize the overall score of the tree. Thus, in choosing between Trees 1 and 2, one would choose Tree 1 to maximize the overall score. Generalizing, one can say that in considering the proto-phoneme /b/ as the root node, one can choose whatever set of grandchildren (and any nodes below grandchildren) that would maximize the overall value of the tree, without fear of loosing better alternatives with the proto-phoneme at the root node.

Now, suppose that when the algorithm comes to consider the candidate proto-

phonemes for the root node in Figure 3.5, it already had information on the situation of the child node. Specifically, suppose it knew, for each candidate proto-phoneme of the internal child node, what the best internal node assignments would be to produce that child node. Then, it could simply try each candidate proto-phoneme for the root node and compare it with the best sub-tree for each candidate for the internal child node (i.e. so that the child has a given node but the rest of the internal nodes of the tree are assigned so as to maximize the value of the tree). This process is captured more formally in Algorithm 1.

Algorithm 1 Exploring a Given Node

combinations make a list of possible combinations of child phonemes
$bestTrees = Null$
for all symbols in the alphabet **do**
  $maxDOCE = 0$
  $bestTree = None$
  for all combos in combinations **do**
    $symbolDOCE = evaluateNode$ (parent and combo)
    if $symbolDOCE > maxDOCE$ **then**
      $maxDOCE = symbolDOCE$
      $maxTree = NodeandChildTrees$
  $appendmaxTreetobestTrees$
return $ExploreNode(ChildrenArrays)$
EndFunction

What remains is to determine how this information can be available as each node in the tree is considered. This is done by the simple expedient of starting from the bottom nodes and working upwards in a basic dynamic programming algorithm. Thus, the algorithm considers the nodes that are right above the leaf nodes first, calculates the set of best trees for each proto-phoneme, stores them, and then can use them when it reaches the node above. The algorithm is complete. Furthermore, the now runs in

polynomial rather than exponential time, as it only considers (number of phonemes in inventory)$^{\text{number of children}}$ options at each node.

3.2.2 The Biuniqueness Criterion and Multiple Runs

This last remaining problem is created by the biuniqueness criterion in Section 3.1.2. As stated in Section 2.2.1, one goal of the reconstruction is to create a one-to-one relationship between distinct correspondences and distinct phonemes, barring the exceptions created by context. The problem is that when the algorithm is reconstructing a particular proto-phoneme, it does not yet have information about other proto-phonemes it has not yet reconstructed and their contexts.

The simplest approach to the issue of biuniqueness would be to consider, when reconstructing any given proto-phoneme, all the information already available to the algorithm — the other phonemes that have already been reconstructed. However, this means that the outcome of the algorithm is strictly dependent on the order in which the correspondence sets are processed. Suppose the algorithm contains two different correspondences [/k/, /ʃ/] and [/k/, /k/], and these two correspondences occurred in similar contexts. If the one with the /ʃ/ were processed first, the algorithm would reconstruct a /k/ there, and thus there would be a pressure to **not** reconstruct a /k/ for the second correspondence. If the order were the other way around, the situation would be exactly reversed. In this situation, it seems reasonable that a /k/ should be reconstructed for the second correspondence and, potentially, a /ʃ/ for the first.

In order to help ensure that such situations are resolved correctly, the algorithm performs the following operation: all the phonemes are reconstructed, the average DOCE of those reconstructions is taken, all the correspondences (including multiple copies of any correspondences that occur in more than one word) are re-ordered randomly, and the reconstruction repeats. This loop continues as many times as the user specifies, and, at the end, the set of reconstructions with the best average DOCE is chosen.

This method, however, rests on the assumption that orderings that lead to bet-

ter overall reconstructions would have higher average DOCEs. The basis for this is illustrated in Figure 3.6.

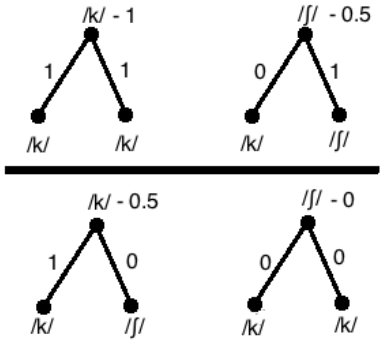

Figure 3.6: Evaluating Correspondences In Two Different Orders

To simplify the argument, it is here assumed without loss of generality, that the *biuniqueness* criterion has a sufficiently high penalty that the same phoneme can **never** be reconstructed for different correspondences, that the only two candidate proto-phonemes in our inventory are /k/ and /ʃ/, and that the only other relevant criterion is majority. While this is a significant number of constraints, they do not in any way alter the overall nature of the problem, except to make the distinctions more clearcut. If, as in the top layer of Figure 3.6, the [/k/, /k/] correspondence is considered first, then the better proto-phoneme for it will be chosen (namely /k/), and the /ʃ/ will be left for the [/k/, /ʃ/]. The average DOCE of those trees is then 0.75. On the other hand, if, as in the lower layer, the [/k/, /ʃ/] is considered first, it may claim the /k/ proto-phoneme, leaving [/k/, /k/] with the worse alternative. However, since that alternative is worse for [/k/, /k/], the reconstruction will have a lower DOCE, and so the average of the trees is only 0.25. Thus, in this case, the algorithm would in fact pick the better solution (i.e. the one in the top layer), since it assigns, overall, better proto-phonemes to the correspondences, and the DOCEs reflect that. Thus, as can be seen, the shuffling method allows the better reconstruction to be the one picked in the end.

Thus, the biuniquiness criterion is within reason satisfied, and the algorithm for

implementing the model presented in section 3.1 is complete.

In this chapter, a model has thus been presented, which, as discussed in Section 3.1 performs better along the chosen heuristics than either the BC or the Oakes model. Furthermore, this section has illustrated that this model can in fact be implemented algorithmically in an effective manner.

The remaining question then, is how well this CMA algorithm performs along the **effectiveness** heuristic, especially as compared to the Oakes and BC algorithms. That question is the topic of the next chapter (Chapter 4).

Chapter 4

Results and Analysis

The goal of this chapter is to test the performance of the CMA at reconstructing proto-phonemes and proto-words, and providing accurate DOCEs. Before addressing that question, however, Section 4.1 explores the user-set parameters of the algorithm (the word sets and the naturality matrices), and then section 4.2 addresses the **effectiveness** of the CMA algorithm.

4.1 Input Data Used

The CMA requires two main types of input data: the word sets in different languages and the naturality matrix (the naturality matrix will here also be used to determine the context classes). The next two subsections are dedicated to discussing the test word sets and naturality matrices used here to evaluate the performance of the CMA.

4.1.1 Word Sets

Three data sets were used to test the CMA algorithm: one is a textbook-based example used to test the properties of the CMA algorithm (Section 4.1.1.1), and the other two are entirely real data sets that have been used by the other algorithms and are used here to compare the CMA's performance to that of the other algorithms (Sections 4.1.1.2 and 4.1.1.3)

4.1.1.1 Textbook Example: Aroma, Hula, and Sinaugoro

The test data set is an example problem from Terry Crowley and Claire Bowern's An Introduction To Historical Linguistics (2010, p.105), where the goal is to reconstruct word forms in the proto-language given word sets in three Central Papuan languages: Aroma, Hula, and Sinaugoro (henceforth, this data set is referred to AHS). There are 82 words from each language, and the same set of meanings is covered by each word list. The tree for this example was derived from the solution sets (provided by Claire Bowern): most of the changes occur in two of the languages (Aroma and Hula), so it was inferred that the tree is as in Figure 4.1, where AH and AHS represent the proto-languages of their child nodes.

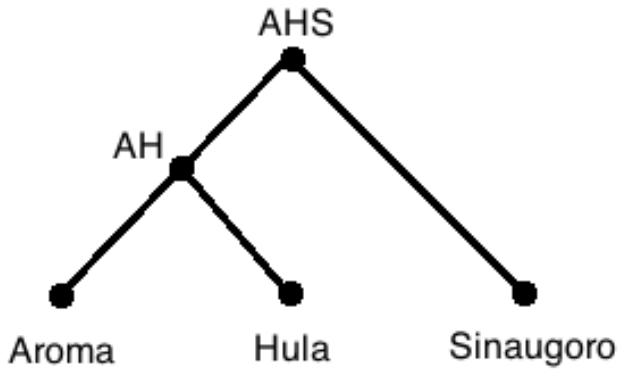

Figure 4.1: Three Possible Trees for Oakes Data

A textbook based example is useful in testing the CMA, because an application of the classic comparative method to the data alone (without any external knowledge) should be able to produce the same reconstructions as the 'answers' to the problem. Such a situation, where the word lists are sufficient to correctly infer the reconstructions, is very rare (Trask 1996) and this makes this example ideal for testing the CMA: the algorithm should be able to find the correct reconstructions, as it has all the necessary input information. Furthermore, Claire Bowern argues that this is a difficult problem

based on real data, and thus a fair test for the CMA (personal communication). In fact, the data in the AHS provides challenges that put to the test most of the different aspects of the CMA model, as described below.

Notably, in this data set, as in all others, a majority of the correspondence sets are entirely straightforward, as in the AHS for word /skin/ shown below in the three languages (with the human reconstruction as th top line):

Table 4.1: Trivial Correspondence Sets

Proto	k	o	p	i
Aroma	-	o	p	i
Hula	k	o	p	i
Sinaugoro	k	o	p	i

The correspondences represented in the last three columns are trivial for both the human linguist and the CMA, as all the phonemes in the correspondence are identical (e.g. [/o/, /o/, /o/]). Only the first correspondence [/Ø/, /k/, /k/] requires any work at all. It is such nontrivial correspondence sets that are of interest. In particular, there are three correspondence patterns that both occur frequently in the AHS data and pose interesting challenges for the CMA:

Pattern 1: [/k/, /k/, /g/] This type of correspondence set occurs in several words, such as in the cognate set for 'voice:' [/karo/, /karo/, /garo/]. This simple correspondence set provides a test to determine whether the CMA can accurately judge the *naturality* of sound changes. The human reconstruction in this case is /g/ (/garo/ for the whole word), as /g/ > /k/ is judged to be a more natural change than /k/ > /g/, in keeping with the theory of Wurzel 1989 that devoicing is more natural than voicing (p.2). As a second argument, /g/ is also the better reconstruction, because /k/ is reconstructed elsewhere for the trivial correspondence set [/k/,/k/,

/k/] (It would have been better to find an example that does not involve context, but no nontrivial examples are available in the data set. Please see Section 4.2.3 for handing of this issue.)

Pattern 2: [/ð/, /Ø/, /Ø/] This type of correspondence set occurs in words such as 'tongue' (/maða/, /maa/, /maa/ in Hula, Aroma, and Sinaugoro respectively). The human reconstruction for this correspondence set is a /ð/, representing the fact that deletion of the /ð/ in Hula and Sinaugoro is considerably more likely than it's insertion in Aroma. This constraint also speaks to the *no empty node* criterion of the CMA.

Pattern 3: [/ɣ/, /ɣ/, /ɣ/], [/ɣ/, /Ø/, /ɣ/], and [/Ø/, /ɣ/, /ɣ/] This last example is the most complex for both the human linguist and the CMA model. The three correspondences occur very frequently in the data set, in words such as 'bowels' (/inaɣe/,/inaɣe/, /tinaɣe/), 'octopus' (/ulia/, /ɣulia/, /ɣulita/), and 'liver' (/ɣae/, /ae/, /ɣate). Two of these occur in complementary distribution (i.e. they always have different contexts): [/ɣ/, /ɣ/, /ɣ/] occurs only intervocallicaly and [/ɣ/, /Ø/, /ɣ/] only at the beginnings of words. The third correspondence set ([/Ø/, /ɣ/, /ɣ/]), on the other hand shares context with both of the other two: the correspondence set occurs word-initially in the example above and intervocalically in words such as 'alive' (/mauli/, /maɣuli/, /maɣuli/). As discussed above, the *biuniqueness* criterion suggests that one phoneme can be reconstructed for both of the correspondences in complementary distribution but that the third correspondence must have a different phoneme. The solution set for the human linguist offers two ways to do this reconstruction. First, a linguist might reconstruct a /ɣ/ for [/ɣ/, /ɣ/, /ɣ/] and [/ɣ/, /Ø/, /ɣ/] and /w/ for [/Ø/, /ɣ/, /ɣ/] (as /w/ does not occur otherwise before high vowels). Alternatively, the linguist might reconstruct /ɣ/ for [/Ø/, /ɣ/, /ɣ], and the other two correspondences as another phoneme such as /ŋ/, which would be useful to balance the resulting sound system

(Bowern 2013). Notably, the second of these solutions requires a reconstruction of a phoneme not represented in the data for the daughter languages on the basis of a criterion (balance of the phonological system) that is not encoded in the CMA. This is a rare example where reconstructing a foreign sound is useful, and it goes against the *no foreign sounds* criterion. This example thus presents the most difficult challenge for the CMA, and it would be considered a positive outcome if the CMA output either of the solutions encoded above.[1]

These three examples represent the main challenges in this dataset for the CMA algorithm. The only criterion not directly tested is *majority*, but it appears in each of the examples (as will be shown later) in creating a heavy bias towards reconstructing proto-phonemes already in the correspondence set. Section 4.2.3 will describe in detail how well each the CMA performs for each of these examples.

4.1.1.2 Comparing Against Other Algorithms: Proto-Malayo-Javanese

The other two data sets were chosen to be the same as those used by Oakes and Bouchard-Côté in testing their algorithms, so that comparisons can be drawn. The first such data set is one used by both Oakes and Bouchard-Côté: four word lists from Sundanese, Javanese, Malay, and Madurese, drawn from Nothofer's 1975 thesis. There are approximately 200 words from each langauge, and, again, each word list covers the same set of meanings, where the meanings are drawn from the 200-word Swadesh list (Nothofer 1975, p.226). The algorithm is meant to reconstruct Proto-Malayo-Javanese (henceforth, referred to as PMJ), and the reconstructions given by Nothofer are taken to be the 'correct' answers by Oakes, BC, and myself. Furthermore, the CMA, like the other two algorithms, adopts the tree given by Nothofer 1975 (p.4). Another convenient aspect of this data set is that the cognate sets in it are hand-labeled. This means that the first part of the CMA algorithm (Mattis List's algorithm for cognate and

[1]For the purposes of the tests in the rest of this paper, the solution used is the one in which /w/ is reconstructed for [/Ø/, /ɣ/, /ɣ/] and /ɣ/ is reconstructed for the other two.

correspondence identification) is used mainly to identify correspondences rather than cognates. Thus, each of the algorithms receives approximately the same input (with cognates labeled the same way) to the 'reconstruction' portion of the algorithm. As a result, when the outputs of the programs are compared, the results differ only based on the differences in the 'reconstruction' portions of the algorithms. This is convenient, since the 'reconstruction' step of the algorithm was the main area where new concepts were introduced for the CMA.

This data set, however, required some pre-processing before it could be used to run any of the three algorithms. I here use the same version of the data used by Bouchard-Côté et al. and accept the pre-processing steps they took. First and foremost, this is convenient as it allows comparisons between the BC, Oakes (as Bouchard-Côté et al. ran Oakes algorithm as well), CMA algorithms that are not affected by differences in input. The main pre-processing step taken with both this and the next data set is that all non-spacing diacritics were removed (Bouchard Cote, personal communication). In this data set, such diacritics seemed to represent only accent marks of some sort, and thus omitting them does not necessarily have an affect on the reconstruction. There may still be a cost, but, if there is, this is a cost the CMA shares equally with the other algorithms, and the decision was made to accept this modified version of the data for the sake of having the same input into the CMA as was used for the Oakes and BC algorithms.

Besides serving as a basis for comparison, this dataset presents its own set of challenges for the CMA. The most obvious issue is that a considerable number of the human reconstructions clearly rely on information outside of the four word lists used as input into the CMA. Thus, for example, for the word 'belly,' a word is only listed for Sudanese and that word is /beuteung/. The CMA (and a linguist given only that information) is likely to reconstruct /beuteung/. The human reconstruction, however, lists /βetteŋ/, which cannot in any way be predicted from the data. Other decisions rely on semantic information that is outside the current ability of the CMA to process. Thus, for

example, the cognate set listed for the word 'right' contains words from three of the languages: [/teŋen/, /kaŋan/, and /kanan/]. The human linguist might reconstruct, and the CMA might be expected to reconstruct, something similar to these forms. Instead, the human reconstruction listed is /benner/, which is also the reconstruction for the word meaning 'correct.' A human linguist might, as Nothofer apparently did, decide that only one word should be reconstructed for these related meanings, but such decisions are far outside the scope of the CMA. Such reconstructions, depending heavily on either semantic data or data not available in the word lists, accounts for approximately 12% of the data. A secondary challenge, similar to the one described above, also arises: inserted final phonemes. This problem appears in data sets such as the word for 'liver,' which provides the four words as [/ati/, /ate/, /hati/, /hate/]. Based on just that information, the reconstruction might be some variation of /hati/ or /hate/. The reconstruction provided by Nothofer, however, is /hatey/, and such final consonants are added 27% of the data.

Overall, this dataset is thus very difficult for the CMA to work with. The main questions with this data is how well the CMA can handle the dataset despite such problems and how well the CMA performs when compared to other algorithms.

4.1.1.3 Testing Scalability: Oceanic Data

The final data set used is meant to test the scalability of the CMA model, and it is one of the data sets used by BC in testing their algorithm. It is a set of 45 word sets covering the same 210 meanings, each from a language that is one of the descendents of proto-Oceanic (henceforth, this data is referred to as Oceanic). The data is drawn from the Austronesian Basic Vocabulary Database (ABVD), but pre-processed in the same way as the previous data set by Bouchard-Côté (Greenhill 2008). The diacritics in this dataset represented a variety of phonetic information, with nasalization, labialization, and accent being perhaps the most common factors represented through diacritics. Notably, these could be useful in revealing correspondences and contexts so deleting

them does impose a potentially significant cost. Nonetheless, using the same input data as the algorithms the CMA is being compared was considered to be of more importance (for the purposes of the comparison!) then retaining the diacritics, and thus they were deleted. The goal of the algorithm is to reconstruct the set of words in Proto-Oceanic, and the answers here are taken to be those provided by Blust (1993) as listed on the ABVD site (Greenhill 2008).

Many challenges arise in using this data set. The algorithm (whether the CMA or BC one) works with 45 languages in the Oceanic group, although the group itself. Furthermore, unlike with any previous data set, it is highly unlikely that Blust, whose reconstructions were used as the example 'human' reconstructions, created his reconstructions based on the same languages as used here for the CMA and BC algorithms. On the one hand, this gives an advantage to the automated reconstruction, which can consider more languages simultaneously than the human linguist usually does. On the other hand, this situation means situations such as the ones described for the PMJ data in Section 4.1.1.2 are far more frequent.

Once these issues are taken into account, the biggest challenge for the CMA becomes the amount of missing data. For many of the words that have human reconstructions, only three or four languages are available to the CMA, which means it might be missing any number of other information that the human linguist may have considered.

This data set, thus, is very difficult for the CMA to work with, and, furthermore, the results cannot always be usefully compared to those of the human linguist (since the human linguist may have been considering different languages). However, it is interesting to consider a large data set such as this one to see how well the CMA can perform despite these significant issues and, furthermore, to determine how well it performs in comparison to the BC algorithm, which faces all of the same problems.

These three data sets form the core input data on which the CMA is tested. The second type of input the CMA requires is the naturality matrix and context classes necessary for the *naturality* criterion described in Section 3.1.2, and the next section is

dedicated to describing the two types of naturality matrices used.

4.1.2 Naturality Matrices and Context Classes

The two naturality matrices used in testing the CMA represent two different approaches to modeling sound change: the classic theory and intuition based approach, and a new endeavor to use purely empirical estimates. The former is represented by Mattis List's sound class model (SCA) (List 2012b), and the latter by the a matrix formed on the basis of the data in the Automatic Sound Judgment Program (ASJP) (Brown et al. 2011).

The other advantages of using these particular databases is that both of them are very general (i.e. are not created for a specific language family) and already converted into something approximating naturality matrix form by Mattis List.[2]

The SCA model of sound change, described in detail in List 2012a, is a sound class model, wherein IPA phonemes are grouped into classes and the cost (the value inside of the corresponding cell of the naturality matrix) of a transition between two sounds is zero if both sounds are in the same class. For calculating the transition cost for two sounds from different classes, List uses a scoring scheme that, as stated in List 2012b, "reflects common sound change processes which are often discussed in the literature" (p.42), combined with List's own "intuition and introspection" about which sound changes are most likely (List personal communication). The SCA model can thus serve as an approximation for the naturality matrix a human linguist might use in reconstructions: it is based on both literary background and intuition, just as that of a human linguist. Notably, however, List encodes in his model the similarity between sounds, so that the scoring scheme assigns equal probability to change in either direction (e.g. score (/b/ > /b/) = score (/b/ > /p/).

The second model, from ASJP, represents a new type of approach — a purely empirical one. Like the SCA, the ASJP is a sound class system, with the cost of

[2]See the .bin files within the lingpy program.

transition within a class set to zero. The method for calculating the cost of transitions between classes, however, is different. Brown et al. 2011 created a database of word sets from "over half the world's languages and nearly all of the world's genetic groups" (p.7). They then used this database to calculate the frequency with which any two distinct sound classes occur in a correspondence. At the end, they argue that they calculate, for every pair of classes, "the pure tendency of sounds [classes] to correspond to each other, independent of how often the individual sounds occur" (Brown et al. 2011, p.8). Furthermore, they claim that their results contain not all but "most of the worldwide inventory of common correspondences along with a representative sample of the rare ones" (Brown et al. 2011, p.8). Notably, this scoring scheme is also assigns equal probability to changes between the same phonemes, as the scoring scheme finds the frequency of a correspondence, rather than a sound change. However, while this is an admitted failing of this scoring scheme, it is also the only easily available empirical naturality matrix, and thus an interesting option to use in testing the CMA.

For the CMA, the matrices were edited slightly to ameliorate their blindness to the directionality of sound change. Specifically, each matrix originally offers a score for each pair of sounds, as in Table 4.2a.

Table 4.2: A sample matrix with symbol set /A/, /B/, /C/

(a): original	A	B	C	(b): adjusted	A	B	C
A	0.8	0.1	0.2	A	0.76	0.048	0.19
B	0.1	0.8	0.7	B	0.032	0.51	0.45
C	0.2	0.7	0.8	C	0.12	0.41	0.47

For the CMA, each row in such a matrix is normalized so that the values in it sum to 1 (as in Table 4.2b). Thus, the value in each cell in row /A/ now reflects a normalized score closer to the probability that /A/ changes into the column of that

cell. For example, in the original matrix (Table 4.2a), /C/ (third row) has a higher similarity scores with /B/ and itself than with /A/ (0.8 and 0.7 are both greater than 0.2). Thus, in the new matrix (Table 4.2b), /C/ changing into /B/ (henceforth, /C/ > /B/) is more likely than /C/ > /A/ (0.7 > 0.12). On the other hand, the similarity score of A and B was even lower than that of A and C (row 1 of Table 4.2a). Thus, /A/, unlike /C/, is more likely to change into /B/ than into /C/ (row 1 of Table 4.2b).

There is, however, one remaining problem with this matrix. In a real naturality matrix covering a majority of the phonemes in the IPA, each row can have over one hundred values and so, when the scores are normalized to add up to 1 as described above, any given entry in the matrix is very small (e.g. 0.003). Such values, in turn, create problems when the overall DOCE of a vertex is calculated, as described in what follows. Recall from Section 3.1.2 that the DOCE of a given internal nodes is calculated via equation (4.1), with a score and a weight for each factor.

$$DOCE = score_{naturality} * weight_{naturality} + score_{majority} * weight_{majority} + ... \quad (4.1)$$

The naturality score is simply an average of relevant cells in the naturality matrix (see Section 3.1.2 for a full description). If this average varies between 0.001 and 0.1 (as it would in a full matrix constructed by the same rules as the matrix in Table 4.2b), then the variations in the naturality score will never be as important as the variations in the majority score, which varies between 0 and 1 (again, see Section 3.1.2 for a full explanation). This situation, however, contradicts the very purpose of assigning separate weights to criteria — namely, that the user should be able to make any of the variables significant. In order to ameliorate this situation, a further manipulation is done on the normalized matrix (as the on in Table 4.2b): the value in each cell is divided by the maximum value across all cells, producing a matrix such as the one in Table 4.3, where all cell entries in Table 4.2b have been divided by the maximum (0.76).

Since this maximum value is still less than or equal to 1 (as the matrix has been normalized), the value of the number in each cell will increase as will the difference

Table 4.3: A Final Naturality Matrix

	A	B	C
A	1	0.063	0.25
B	0.042	0.67	0.59
C	0.16	0.54	0.62

between two cells in a given row. This larger difference, in turn, allows *naturality* to play a greater role in equations such as (4.1). This manipulation retains and does not affect the directionality of the changes (since all values were multiplied by the same entity, all ratios are conserved). On the other hand, the normalization factor (that all rows sum to one) is lost. This normalization, however, is not a significant part of the model, as no other elements of the model allow the CMA to generate the probability of a given reconstruction — the CMA only provide a score; this type of naturality matrix, then, fits into the overall CMA model and captures the directionality required.

This methodology is far from perfect, but it does provide a practical way of adding directionality to the naturality matrices used here, while retaining the overall usefulness of the naturality matrix.

The final question concerns context classes. Recall from the description of the *biuniqueness* criterion in section 3.1.2, that the user defines a set of classes for context, so that all the phonemes in the same class count as one 'context.' For each of the naturality matrix models above, the most natural set of classes is simply the set of sound classes used by the model. This way, the scores and penalties assigned for the *biuniqueness* criterion and those assigned for *naturality* will be consistent in what they consider to be a class. Thus, the sound classes of each naturality model are also used

to represent the context classes when that naturality model is used.

At this point, all the necessary input to the CMA has been described, and the next section is dedicated to discussing the performance of the algorithm.

4.2 Testing the CMA

4.2.1 Evaluating Results

The first task is devising a method for evaluating the results of the algorithmic output. Here, I adapt the "Levenshtein distance" method used by Bouchard-Côté et al. 2013, for ease of comparison with their and Oakes' results (p.3). The Levenshtein distance is the number of substitutions, deletions, and insertions required to change the 'correct' answer (i.e. the reconstruction of a proto-word given by the human linguist) into the reconstruction provided by the algorithm (Bouchard-Côté et al. 2013). Each operation (substitution, deletion, and insertion) can have a cost, and I here use the same costs as Bouchard-Côtét et al: the cost of each operation is set to 1 (Bouchard-Côté, personal communication). The quality of a run of the algorithm is thus the average of the Levenshtein distances between the human and algorithmic reconstructions for each word.

There is, however, one more potential problem. Even though most of the words in the data sets used are hypothetically cognate, some of them are sufficiently different phonetically that reconstructing a single proto-form from all of them would be impractical. Moreover, in some cases, they are not in fact cognate at all. In the test AHS dataset, there is precisely one such example, where a single semantic meaning is broken up into two cognate groups: the words for 'to die' in the three languages are [/mate/, /wareɣa/, /k^{w}area/]. Furthermore, a second word is listed for Aroma — /mae/ (Crowley and Bowern 2010). It is clear that /mate/ and /mae/ are cognate, as are /wareɣa/ and /k^{w}area/, and that a single proto-word should not be reconstructed from these four words. In fact, the solution set offers two reconstructions for 'to die.' In order

to deal with such situations, the 'cognates and correspondences' algorithm is run on the cognate sets in the input and allowed to separate these cognates sets into potential further sub-groups. The 'reconstruction' portion of the CMA is then applied to these subgroups. While this can potentially create unnecessary splits of cognates, it performs effectively: on the AHS data set, for example, the only split into sub-groups occurs at the aforementioned word set, which is the exactly correct response. However, for both the PMJ and the Oceanic data sets, only a single reconstruction is provided for each meaning, yet more than one proto-word may have been reconstructed for a given meaning, based on the subgrouping at the 'cognates and correspondences' step. The solution taken here is to use the reconstruction that is the smallest Levenshtein distance away from the correct solution. The justification for this is that if two alternative proto-words were reconstructed by the algorithm (e.g. /mate/ and /kareɣa/ in the example above), then the correct way to evaluate the reconstruction for this proto-word is to compare the human answer to the algorithmic reconstruction that captures the same general idea as the human reconstruction (e.g. if only /mate/ was listed in the answer, the CMA reconstruction /mate/ should be used for comparison). The best way to pick the right algorithmic reconstruction, in turn, is to choose the one closer in Levenshtein distance to the human reconstruction (/mate/ is closer to /mate/ than /kareɣa/ is). Nonetheless, this might create some slight bias in the comparisons between the performance of the CMA algorithm and that of other algorithms.

4.2.2 Selecting Naturality Matrices, Criteria Weights, and Trees

The next issue is that of choosing the naturality matrix, selecting trees, and assigning weights to the various criteria.

4.2.2.1 Weights for Criteria

The first task, as it bears on both of the others, is the choice of weights for the various criteria. For the experiments in this paper, the weights chosen were as shown in

Table 4.4.[3]

Table 4.4: weights for Algorithm

criterion	weight
biuniqueness	0.1
naturality	0.45
majority	0.35
no Empty Node	0.1

The logic behind this assignment is a combination of theory, praticality, and experiment.[4] *Naturality* is perhaps the most important consideration, as listed in Section 3.1.2, and it is a reasonably reliable criterion. *Majority*, as a crude but also important criterion, follows after naturality. *Biuniqueness*, as stated in Section 3.1.2 is also a very important criterion. However, the sound class context implementation of this criterion used by the CMA is not necessarily ideal (e.g.it considers two different phonemes within the same sound class to be the same context), and thus its role is significantly diminished. And, finally, the *no Empty Node* practical criterion has been set to 0.1. This makes it less significant than both *majority* and *naturality*, suggesting a preference but not a requirement against reconstructing empty nodes. As described in Section 3.1.2, the *no Empty Node* criterion merely intensifies the existing bias in the naturality matrices against reconstructing empty phonemes. The specific value of 0.1 was chosen experimentally. These weights and reasoning, thus, are simply an example

[3]Certain specific experiments in this section used slightly different values, but, if any different values are being used in an experiment, this will be stated in this paper.

[4]Experiments were performed to confirm the usefulness of individual criteria, as well as combinations of criteria. To some extent the usefulness of specific criteria depends on the data set. For example, naturality is more likely to be useful when the naturality matrix is more applicable to a given data set. However, due to space constraints, I do not go into the details of such experiments here.

of how the weights can be used. If the user has particular information, for example, on how well their naturality matrix fits their data, they may choose different weights. Finally, each data set was run 10-20 times (10 for the larger data set), and the run with the best average DOCE was picked (recall from Section 3.1.2 that runs may differ as correspondence sets are shuffled and this affects biuniqueness calculations).

4.2.2.2 Naturality Matrices

As the AHS dataset offers the least noisy data (being a textbook example), it was used to choose between the two matrices discussed in Section 4.1.2. The AHS dataset, with the criteria weights discussed above, was run 500 times with the ASJP matrix and 500 times with SCA matrix. For each such run, the average Levenshtein distance between the human and the CMA reconstructions across all reconstructed words was calculated. Since the *biuniqueness* criterion had a non-zero value, there was a randomness factor in these averages (see Section 3.2.2), and these values created a distribution for each matrix, on which the t-test scores were based.

The results indicated that the ASJP matrix is the better matrix for the *naturality* criterion. The average levenshtein distance across all the runs was 0.25 for the ASJP matrix and 0.28 for the SCA matrix. This difference, while small, proved statistically significant with a p-value $< 1.76 \cdot 10^{-84}$.[5]

As a very preliminary hypothesis, this would suggest that the data-based ASJP matrix captures more general trends than the SCA matrix, which is an encouraging result for any ongoing efforts at creating sound change databases. Another preliminary hypothesis may be that both matrices capture the basic trends in sound changes, since the results with both matrices are very good. Using only this data set, however, confirms

[5]There does not seem to be a consistent set of phonemes that the ASJP reconstructs correctly and the SCA reconstructs wrongly. This may be in part because one can only observe one run at a time with each matrix, and individual runs are affected by the randomness created by that particular runs' shuffling of correspondence sets. As such, it was not productive to give examples of the differences between the results provided by the two matrices.

only that the sound changes in this reasonably simple data set are well captured by both matrices. To confirm both of the hypotheses, it would be useful to run both these and other matrices on a greater number of maximally noiseless data sets where the results can be analyzed.

4.2.2.3 Trees

As stated in Section 4.1.1, the trees considered by human linguists to be correct were used for all three data sets. The goal of this section is to justify this choice, by illustrating for the transparent AHS data both that the 'correct' tree performs better and why that might be the case.

In approaching this problem, three possible trees were considered: the 'correct tree' (Figure 4.2a), a flat tree such as Oakes uses in his algorithm (Figure 4.2b), and a 'wrong tree' where languages are grouped together differently than a human linguist would group them (Figure 4.2c).

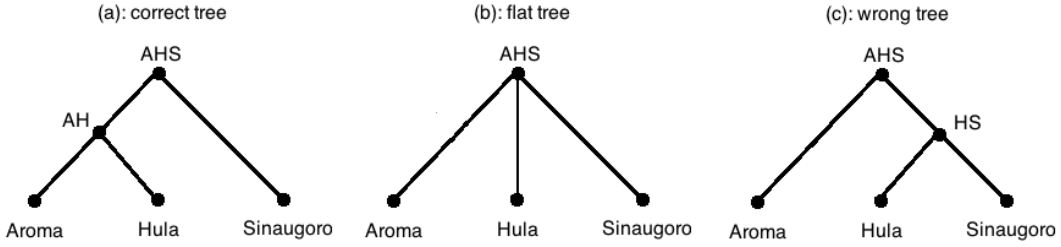

Figure 4.2: Three Possible Trees for AHS Data

Next, the quality of the reconstructions produced when using each tree was tested. 200 runs were done with each tree, using the criteria weights listed in Section 4.2.2.1 and the ASJP matrix. The results, as can be seen in Table 4.5 confirm the hypothesis: the correct tree performs better than both the flat tree and the wrong tree.

The next question then is why the right tree performs consistently better. Recall from Section 4.1.1.1 that one of the simplest examples in the AHS data was the correspondence set [/k/, /k/, /g/] from the cognate set for 'voice.' This example will be used

Table 4.5: Performance of Different Trees

tree type	ave. Lev. Dist.	t-test
right tree	0.27	-
flat tree	0.45	0
wrong tree	0.29	0

here to illustrate the type of inferences made correctly for the good tree, but incorrectly for the other two trees. As I mentioned in Section 4.1.1.1, the *biuniqueness* criterion is also involved in this particular correspondence set. To avoid any issues that may be caused by that, the correspondences and reconstructions below were taken from runs with the three trees where *biuniqueness* was set to 0. Since the *no Empty Node* criterion is irrelevant in this case (as there are no /Ø/ in the correspondence set), that was also set to 0. Thus, the correspondence sets and their reconstructions below are drawn from a run where the only criteria were *naturality* set to 0.6 and *majority* set to 0.4.

Consider the four trees in Figure 4.3. Figure 4.3a represents the reconstruction chosen by the CMA for the example correspondence set on the 'correct' tree (both the internal and the root node reconstructions are presented). Figure 4.3b shows the same for the flat tree. Finally, Figures 4.3c and 4.3d show two different possible reconstructions on the wrong tree, of which Figure 4.3c is the one chosen by the algorithm. For each tree, the naturality score of a transition between the parent and the child nodes is labeled along each branch. Each node, furthermore, is labeled with its majority score (m) and its naturality score (n). These scores are calculated as averages of the majority and naturality scores along each branch (as discussed in Section 3.1.2). Finally, at the bottom of each tree, the overall DOCE of the tree is calculated, using Equations (3.2)

and (3.1), as discussed in Section 3.1.2. Combined, these trees illustrate why using the correct tree can be critical to achieving the right reconstruction.

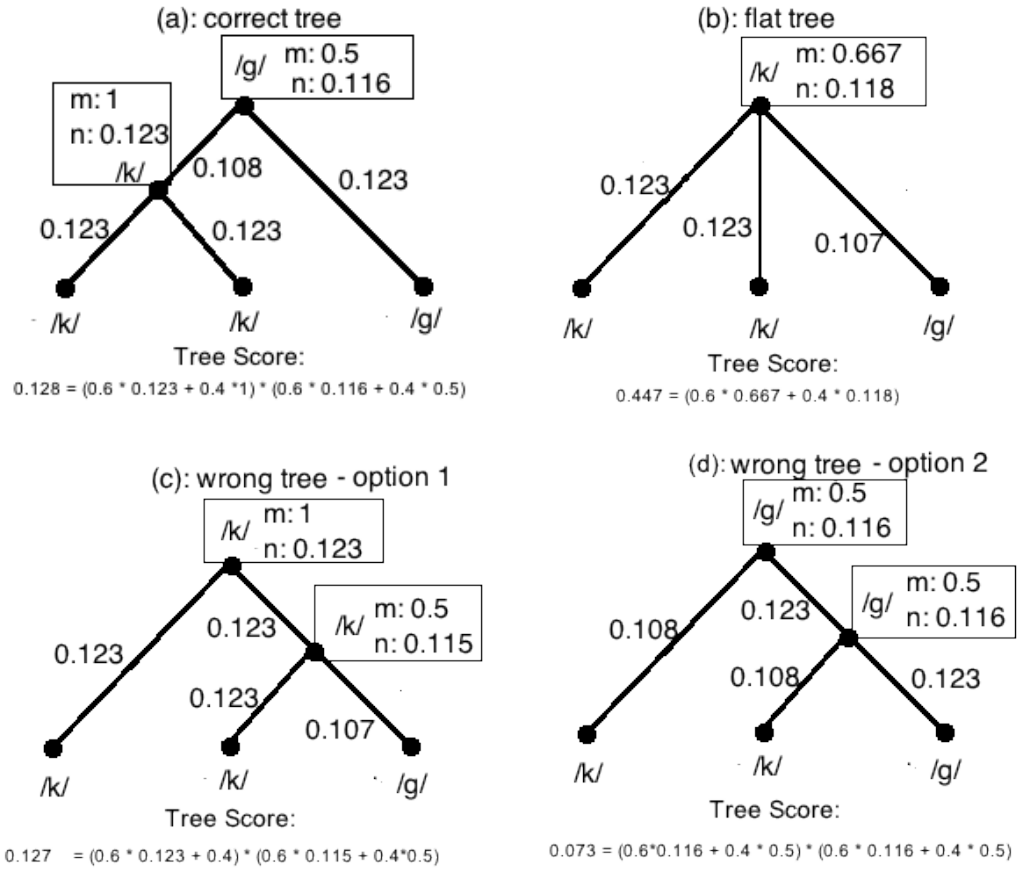

Figure 4.3: The [/k/, /k/, /g/] Correspondence On Different Trees

First, note that the phoneme reconstructed by the CMA for the root node in both the flat tree (Figure 4.3b) and the wrong tree (Figure 4.3d) is not the same one the human reconstructs, whereas the correct tree (Figure 4.3a) ensures that the CMA chooses the same proto-phoneme as the human.

Before addressing the problem trees, it is worthwhile to examine why the correct tree reconstructs /g/ as the proto-phoneme. Notably, the fact that Hula and Aroma are grouped together means that the two languages only have one 'vote' (for both *majority*

and *naturality*) in assigning a score to a candidate proto-phoneme at the root. Thus, the *majority* score of /g/ is 0.5, and the *naturality* score is the average of 0.108 and 0.123. If /k/ were the reconstructed proto-phoneme, the *majority* score would be the same, but the *naturality* score would be the average of 0.123 (on the left branch) and 0.107 (on the right branch), yielding a score of 0.115. Thus, /g/ is correctly chosen as the proto-phoneme. Notably, the CMA thus correctly deals with at least this instance of one of the three correspondence patterns in the AHS that are examined here (see Section 4.1.1.1)

The problem with using the flat tree is the easiest to explain. Simply put, the Aroma and Hula languages influence the choice of the proto-phoneme too significantly. First, they contribute two 'votes' to the *majority* criterion, so that if the CMA chooses /k/ as the proto-phoneme, the *majority* score is (2/3), and if the CMA chooses /g/, that score is only (1/3). In the correct tree, the two languages together only contribute one vote to the majority calculation for the root vertex, through the internal AH node (see score calculations in Figure 4.3a). A similar situation occurs with naturality. (Throughout the discussion of naturality, see Table 4.6, which represents the relevant subset of the ASJP naturality matrix as adapted for the CMA.)

Table 4.6: Naturality Matrix For [/k/, /k/, /g/] Example

	/k/	/g/
/k/	0.123	0.107
/g/	0.108	0.123

As seen in the Table 4.6, the naturality of a phoneme remaining the same is higher than that of it changing. Thus, in this case also, Aroma and Hula having two 'votes' for naturality in the flat tree increases the DOCE of /k/ as the reconstructed proto-phoneme. If, on the other hand, /g/ was the proto-phoneme, the naturality score would

be $(0.108+0.108+0.123)/3$, which is considerably less than the $(0.123+0.123+0.107)/3$ that becomes the naturality score when /k/ is used as the proto-phoneme. Thus, the case of the flat tree becomes clear: allowing languages that should be grouped together to contribute directly to the evaluation of the proto-node means that those trees have significantly more influence than they should over the eventual outcome.

The case of the wrong tree is slightly more complex, and, here, two trees are presented: one where /k/ is reconstructed as the proto-phoneme (Figure 4.3c) and one where /g/ is reconstructed (Figure 4.3d). The task here is to determine why the tree with the proto-phoneme /k/ receives the higher score. The basic answer, as before, is that the split up language subgroup (Hula and Aroma) has too much influence. Consider the internal node that is the parent of Hula and Sinaugoro. At that node, by itself, reconstructing /k/ receives a lower score than reconstructing /g/, as shown in Figure 4.3c and Figure 4.3d respectively. Majority is indifferent here, and *naturality* criterion prefers /g/ > /k/ over /k/ > /g/, which fits with the notion that devoicing is more natural than voicing (Wurzel 1989). However, this is insufficient to make /g/ the better reconstruction at the root. In comparing the internal node in Figures 4.3c and 4.3d, note that reconstructing /g/ at the internal node is just barely more advantageous than reconstructing /k/. On the other hand, reconstructing /k/ at the root node given that a /k/ has been reconstructed at the internal node (as in Figure 4.3c) is considerably better than reconstructing /g/ at the root even given /g/ at the internal node (as in Figure 4.3d). This holds both for *majority* and *naturality*. The reason for this is that the Aroma-Hula subgrouping, which already contributed to the score of the internal node to make the gap between the scores of internal node /g/ and internal node /k/ smaller, now also contributes to the score of the root node. Thus, the Aroma-Hula subgrouping contributes to the scores of two nodes, where it should only be able to contribute to one node. In conclusion, splitting up languages meant to be subordinated to a single node (whether by creating a flat tree or spreading these languages over different subgroups), gives the subgroup more influence over the final reconstruction, leading to

flawed reconstructions such as the one in Figure 4.3c.

The only remaining question concerns the difference in the performance of the wrong tree (average Levenshtein distance of 0.29) and the flat tree (ave. Lev. dist. of 0.45). The main reason for this difference is that a wrong, yet complex, tree might still capture some of the complexity of the situation correctly (e.g. the correspondences in the Hula language should not influence the proto-phoneme directly), and this partially captured complexity may allow for correct reconstructions in certain cases (e.g. when all three phonemes in a correspondence are different.

Overall, the established pattern is thus that the correct tree (i.e. the tree used by the human linguist) provides the best results for the CMA. Confirming this hypothesis is reassuring. This confirmation means that allowing the CMA to involve complex trees does in fact increase **effectivenesss** signifiantly, even though it complicates the model slightly.

4.2.3 The Quality of the Algorithm's Reconstructions

4.2.3.1 Results for the AHS Data

On the AHS data, the test data set, the CMA achieves an average Levenshtein distance of 0.27. On a set of ten runs (recall from Section 3.2.2 that the *biuniqueness* criterion introduces randomness into the model), the algorithm chooses the run with the highest DOCE, which in this case corresponded to a run with an averaged Levenshtein distance of 0.253. This means that 79% of the words are reconstructed exactly correctly and 98% of words have at most one mistake. Overall, the CMA thus outputs reconstructions that are highly similar to the human reconstructions.

The next interesting question, however, is to see how well the CMA dealt with the three patterns identified in Section 4.1.1.1, as well as identify the existing mistakes and determine their source. After a brief discussion of the general approach to mistakes in the CMA, the rest of this section will be dedicated to discussing each of the three patterns (each of which, as it turns out, has a mistake).

First, it is interesting to determine how mistakes should be addressed in evaluating this solution set. Each mistake in the solutions can be described in two ways. A mistake can be considered any 'wrong' phoneme in a word reconstructed by the CMA that causes the distance between the CMA reconstruction and the human reconstruction to increase. Thus, for example, the CMA reconstruction for the word 'octopus' is /ɣulita/, whereas the human reconstruction is /wulia/. The two wrong phonemes are then /ɣ/ (which had to be substituted for /w/ in calculating the Levenshtein distance) and /t/ (which had to be deleted). These two mistakes result in the Levenshtein score of 2 between these two words. However, each CMA-reconstructed phoneme is reconstructed from some correspondence set, and this correspondence set *should* have yielded the a different phoneme (or a /Ø/). A mistake is then any time the wrong phoneme was reconstructed from a correspondence set. Looking at the errors in this second way, two mistakes can be considered the same (in some sense) when the phoneme that was reconstructed and the phoneme that should have been reconstructed are both the same. For example, the CMA reconstructs /ɣunu/ instead of /wunu/ for the word 'breadfruit,' and this word pair exhibits the same mistake as the /ɣulita/ vs. /wulia/ example above.

Notably, there are only three distinct (not the same) mistakes made by the CMA, and they (and their frequencies) are reflected in Table 4.7.

Table 4.7: CMA Mistakes on AHS Data

human	CMA	freq
/b/	/p/	8/17
/Ø/	/t/	1/17
/w/	/ɣ/	8/17

Each of these mistakes has its own explanation and falls into its own pattern, so what follows offers a discussion of each of the patterns along with an analysis of the

mistake that falls into that pattern.

4.2.3.1.1 Pattern One and the /b/ — /p/ Error Recall from Section 4.1.1.1, that there were three types of interesting correspondence sets in the AHS data set. The first of these was the [/k/, /k/,/g/] pattern. As illustrated in Section 4.2.2.3, the CMA deals with the [/k/, /k/, /g/] correspondence that falls into this pattern entirely correctly. The correspondence set [/r/, /r/,/d/] also falls into this pattern, and the CMA also deals with both instances of it correctly. Notably, in both cases there is evidence from the *biuniqueness* criterion in favor of the human reconstruction, but the CMA reconstructs the same proto-phoneme as the human even when the *biuniqueness* criterion is set to 0.

There is, however, a third correspondence set that falls into this pattern, and the CMA is not able to handle this third correspondence set correctly. The correspondence set is [/p/, /p/, /b/] and the human reconstruction (and the one expected for the CMA) is /b/.

This expectation stems from two sources: first, /b/ > /p/ should be more natural than /p/ > /b/, at least in keeping with the theory that devoicing is more natural than voicing (Wurzel 1989). The second is that the proto-phoneme /p/ is reconstructed for another correspondence set: [/p/,/p/, /p/].

Nonetheless, in eight of the thirteen cases that this correspondence set appears, the CMA incorrectly reconstructs /p/, rather than /b/. Interestingly, this violates *biuniqueness*, since the CMA thus reconstructs two different phonemes for the same correspondence set (context does not prove relevant here from a human linguist's point of view). The discussion below illustrates how both reasons for reconstructing /b/ for this correspondence set fall apart in the CMA, and the important weakness of the algorithm that is thus revealed.

The first issue is that the ASJP naturality matrix used for these experiments actually sets the naturality of /p/ > /b/ higher than the naturality of /b/ > /p/ (0.120 vs.

0.119). The naturality of either phoneme staying the same (/p/ > /p/ and /b/ > /b/) is held constant in the matrix at 0.123 (and thus is the same as in Section 4.2.2.3). Given this naturality matrix, the reconstruction (excluding *biuniqueness* for the moment) proceeds exactly as described for [/k/, /k/, /g/] in Section 4.2.2.3 and as reproduced below in Figure 4.4

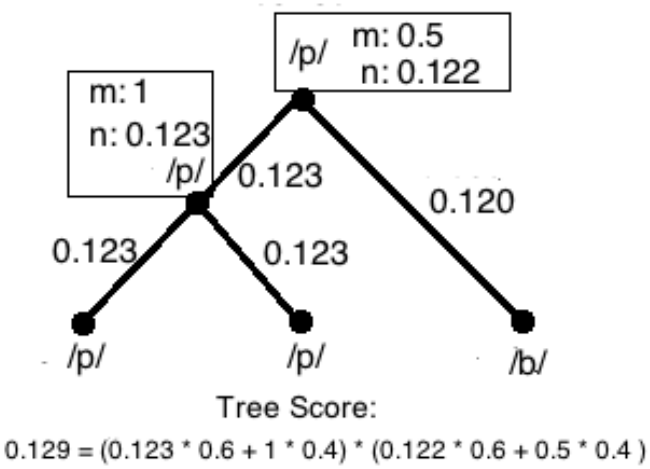

Figure 4.4: The [/p/, /p/, /b/] Correspondence Set Without *Biuniqueness*

Since the naturality matrix does not reflect the naturality principle involved here (i.e. /b/ > /p/ is more likely than /p/ > /b/), the results do not match the human results. This issue with the naturality matrix stems, most likely, from the fact that, as discussed in Section 4.1.2, directionality was introduced somewhat artificially into this ASJP-based matrix. One useful further step would be to try out matrices which have some form of sound change directionality encoded even before pre-processing stages.

There is, however, another piece of evidence that should allow the CMA to reconstruct /b/ for the [/p/, /p/, /b/] correspondence set: /p/ is already unambiguously claimed by the [/p/, /p/,/p/] correspondence set, and *biuniqueness* requires that some other proto-phoneme be reconstructed for [/p/, /p/, /b/] (provided the contexts are the same). Thus, in the instances where *biuniqueness* applies, /b/ will in fact be reconstructed, as shown in Figure 4.5. (n.b. In this figure, the *biuniqueness* penalty is set to

be 0.5. This is an arbitrary value, as the actual penalty is determined, as discussed in Section 3.2.2, by the user and by whether the context is two-sided or one-sided)

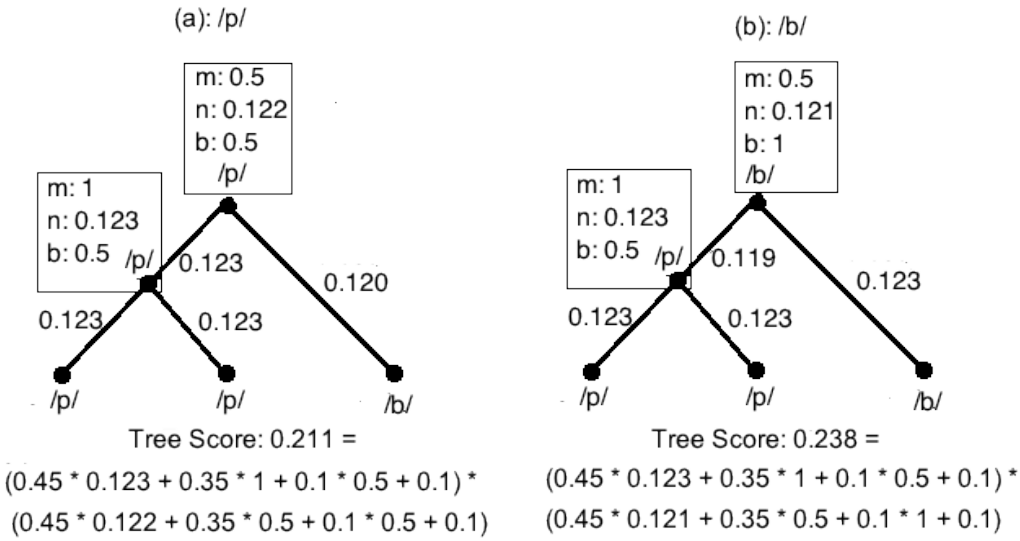

Figure 4.5: The [/p/, /p/, /b/] Correspondence With *Biuniqueness*

In this figure, all four criteria are included, and the weights are assigned as described in Section 4.2.2.1. (The only scores not listed in the boxes next to each reconstrution are the scores for the *no Empty Node* criterion, as its score is always at 1 here, owing to the lack of need to reconstruct /Ø/). As can be seen in the tree scores, adding the *biuniqueness* constraint makes it so that /b/ (Figure 4.5b) scores high as a candidate proto-phoneme than /p/ (Figure 4.5a). Given this, the next question is why the CMA still makes an error in these reconstructions.

Recall from Section 3.2.2, that the CMA goes through multiple runs, and, at each run, all of the correspondences are shuffled (with each occurrence of the correspondence set counting separately). There are thirteen instances of the [/p/, /p/, /b/] correspondence set in the data (12 words in which the correspondence set occurs, with one word having it twice). On the other hand, there are only two instance of the [/p/, /p/, /p/] correspondence set. This means that, on any given run, several instances of [/p/, /p/,

/b/] are likely to appear before any instance of [/p/, /p/, /p/] have been seen, and thus the *biuniqueness* criterion will not be useful in avoiding reconstructing /p/ for those instances. One would assume, however, that there might be a lucky run in which a [/p/, /p/, /p/] occurs before any [/p/, /p/, /b/], that run would be chosen by the algorithm as having the highest average DOCE (again, see Section 3.2.2), and the issue would be averted. Unfortunately, there is a further factor making the situation difficult. To the human linguist, [/p/, /p/, /p/] and [/p/, /p/, /b/] occur in the same contexts, and thus the *biuniqueness* criterion affects them. For the algorithm, however, the context of an *already seen* [/p/, /p/, /p/] must match the context of the [/p/, /p/, /b/] being considered in order for the *biuniqueness* criterion to apply. Given this even more stringent requirement, it is not surprising that, even on a very good run, only five of the 13 [/p/, /p/, /b/] correspondence sets were affected by the *biuniqueness* criterion. Thus, the second type of mistake (and one of the two most frequent ones) is brought about by the implementation of the *biuniqueness* criterion. This gives a first hint that re-implementing the *biuniqueness* criterion may be a useful goal for CMA purposes.

On the other hand, aside from the difficult [/p/, /p/, /b/] correspondence set, the CMA does capture the second kind of pattern correctly.

4.2.3.1.2 Pattern Two and the /Ø/ — /t/ Error The second type of pattern is exemplified by the [/ð/, /Ø/, /Ø/] correspondence set. This correspondence set occurs in four different words, and the phoneme the human (and the CMA) reconstructs is always /ð/. A similar pattern usually holds with the correspondence set [/Ø/, /Ø/, /t/], which appears in 15 words (e.g. 'eye,' 'egg,' and 'milk') . In 14 of these words, both the human linguist and the CMA reconstruct /t/. However, in the word for 'octopus,' the human linguist reconstructs /Ø/ for the same correspondence set, whereas the CMA still reconstructs /t/.

The /Ø/ reconstruction is interesting because it violates the *biuniquenes* criterion. The human linguist seems to reconstruct two different phonemes for the same corre-

spondence set in the same context. Here, the only likely explanation is that there is some external information that the human linguist is using. In that case, this error reflects one of the general problems with automating the comparative method — it is impossible to transmit to an algorithm the wealth of data that a linguist uses in reconstruction. Nonetheless, this particular mismatch between the human and the CMA reconstructions does not, at least, reflect poorly on the ability of the CMA to make plausible inference from the data given.

On the other hand, one can conclude from this discussion that the CMA successfully deals with the [/ð/, /Ø/, /Ø/] pattern of correspondences, which means, among other things, that the *no Empty Node* criterion is functioning correctly. Notably, protophonemes with these correspondences are not reconstructed correctly when *no Empty Node* is set to 0, which illustrates the necessity of the criterion.

4.2.3.1.3 Pattern Three and the /w/ — /ɣ/ Error Recall from Section 4.1.1.1 that this last pattern revovled around three correspondences: [/ɣ/, /ɣ/, /ɣ/], [/ɣ/, /Ø/, /ɣ/], and [/Ø/, /ɣ/, /ɣ/]. [/ɣ/, /ɣ/, /ɣ/] and [/ɣ/, /Ø/, /ɣ/] occur in complementary distribution, whereas the third correspondence set ([/Ø/, /ɣ/, /ɣ/]) occurs in the same context as the other two. Thus, by the *biuniqueness* criterion, a single phoneme can be reconstructed for the first two and some other phoneme must be reconstructed for the third correspondence set. The human solution set offers two different ways of addressing this problem, but this level of sophistication turns out to be irrelevant for the mistake made by the CMA: the algorithm consistently reconstructs /ɣ/ for all three correspondence sets.

The underlying reason for this is fairly simple: the *biuniqueness* criterion (weight = 0.1) cannot override majority (weight = 0.35) when majority is set to (2/3). See Figure 4.6 for an illustration of this with the error /ɣ/—/w/ error.

In this case, the only relevant criteria are *majority* (m) and *biuniqueness* (b). *No Empty Node* is irrelevant, as no such reconstructions are attempted and the score will

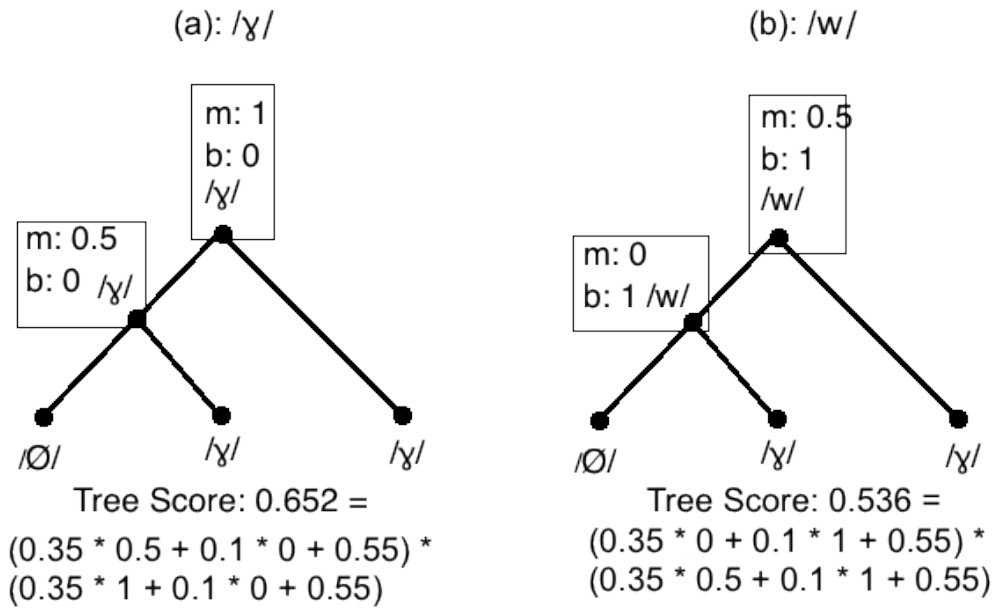

Figure 4.6: The [/p/, /p/, /b/] Correspondence With *Biuniqueness*

always be 1. *Naturality* is also irrelevant, as it will also prefer reconstructions where the sole phoneme of the correspondence set is not changed: /ɣ/ > /ɣ/ will always be higher than /w/ > /ɣ/. If a reconstruction of a foreign phoneme were attempted (as is true in the other human solution discussed in Section 4.1.1.1), the *no Foreign Phoneme* criterion would also be relevant. But, for this correspondence set, *majority* alone is sufficient to demonstrate the issue, so it is assumed that all other criteria have a score of 1. Furthermore, it is assumed that the context is identified correctly and penalized maximally (hence, the *biuniqueness* score is 1). However, none of these assumptions, are sufficient to make a proto-phoneme candidate such as /w/ (see tree in Figure 4.6b) compete with the majority-recommended reconstruction of /ɣ/ (Figure 4.6a). Thus, there is no way, with the current weights, for the CMA to capture the intuition behind the third pattern.

The seemingly simple solution to this issue would be to increase the weight of the

biuniqueness criterion. This, however, is not a viable option, given the way context is implemented in the current model of the CMA. To take the simplest example, if the weight of *biuniqueness* was raised sufficiently to win over majority, the two correspondences in this pattern ([/ɣ/, /ɣ/, /ɣ/] and [/ɣ/, /Ø/, /ɣ/]) that are in complementary distribution according to the linguist, would be considered (in some cases) to have the same context, and *biuniqueness* would require different phonemes to be reconstructed for them. This occurs because, in this case, the context classes defined (the same ones as in the ASJP as discussed in Section 4.1.2) are too fine grained for this correspondence set. Thus, the linguist would argue that [/ɣ/, /Ø/, /ɣ/] occurs only #_ and [/ɣ/, /ɣ/, /ɣ/] only V_V, but the CMA might say that they both occur _i (e.g. in the cognate set for 'hand' and the cognate set for 'lullaby'). Thus, this solution is not feasible, nor, unfortunately, are there any other solutions that do not require changing the implementation of the CMA.

Thus, this final pattern cannot be captured by the CMA model, due to the current implementation of the *biuniqueness* criterion. The combination of the evidence from this pattern and from pattern two above suggests that one of the most important ways of developing the CMA would be improving the *biuniqueness* criterion implementation. There are two significant improvements that could be made to allow the CMA to capture both of the patterns it is now struggling with. First, some way of looking simultaneously across all relevant correspondences for potential *biuniqueness* conflicts would help significantly ameliorate the issue with pattern two. Even more importantly, however, some algorithm must be developed for defining context more accurately for the *biuniqueness* criterion. A variety of steps could be taken in this direction, the simplest of which would be allowing a multitude of different types of context classes (e.g. both 'V_V' and 'i' should be plausible contexts). Ideally, the CMA should then have some method of looking across all instances of a given correspondence (e.g. all the cognate sets in which [/ɣ/, /ɣ/, /ɣ/] occurs and the contexts of each occurrence) and identify what type of context class, if any, all of the instances share. If some variation on such

a model could be implemented, all of the remaining errors on the AHS data would be solved.

Stepping back from *biuniqueness* however, the overall review of the patterns in the AHS data and the few errors made by the CMA suggests that the CMA is generally capable of reconstructing the same proto-phonemes as a human might, so long as all of the relevant information is encoded in the data. As the three Sections above illustrated (4.2.3.1.1, 4.2.3.1.2, and 4.2.3.1.3), the CMA reconstructs the proto-phonemes it would be expected to given the model and algorithm described in Chapter 3, and these phonemes match the human linguist's reconstructions (at least on the AHS data), unless *biuniqueness* issues intervene.

Given these encouraging results, the next task is to compare the performance of the CMA to the performance of the two other algorithms: those developed by Oakes and Bouchard-Côté et al.

4.2.3.2 Results for PMJ and Oceanic Data Sets

The next question, then, is how well the CMA performs in comparison to the two other algorithms available: Oakes and BC. The PMJ data set is the only one to which all three algorithms have been applied, and the respective average Levenshtein distances are shown in Table 4.8.

Table 4.8: Average Levenshtein Distances for PMJ Data Set

model	Ave. Lev. Dist.
BC	1.86
CMA	2.15
Oakes	2.33

Notably, the CMA performs significantly better than Oakes' algorithm, but worse

than the BC algorithm. It was, unfortunately, not possible to determine the statistical significance of the differences, due to a lack of any distribution information from either Oakes or BC for this data set. A very similar situation occurs when the CMA is run on the Oceanic data set, the one used to judge scalability. While Oakes' algorithm was not used on the Oceanic data (as it does not scale well), the CMA achieved an average Levenshtein distance of 3.0175, while the BC algorithm obtained an average Lev. dist of 1.90. This difference here is in fact statistically significant with a $p < 0.0001$.

Three question arise: (a) why is the CMA performance on the PMJ data this much worse than its performance on the AHS dataset, (b) why does the CMA perform better than Oakes' algorithm, and (c) why does CMA perform worse than BC.

With regards to the first question (a), the most significant issues here are concerned with the data itself, as described in Section 4.1.1.2. First, about a quarter (for the PMJ data) of the human reconstructions rely to some degree on information not provided to the algorithm. For the Oceanic data, this number is likely higher. While this is normal for real data, it was not the case for the AHS dataset, and this explains part of the difference. Moreover, the pre-processing step that deleted all diacritics could have effectively masked imported distinctions that would have played into *majority* and *naturality* calculations. In short, the worse performance of the CMA on the PMJ data, as compared to the AHS set, seems to be mostly a function of dealing with real data.

In comparing the performance of Oakes' algorithm to the performance of the CMA, the most likely reasons for the difference are the complex tree structure used by the CMA, the much more fine grained naturality matrices, and the attempt to pay some attention to the *biuniqueness* criterion. Unfortunately, however, as Oakes does not provide reconstructions, it is difficult to do more than guess.

Finally, in comparing the performance of the CMA to that of the BC algorithm, several issues become apparent. The first and one of the most important is that the *no Empty Node* criterion of the CMA, though quite necessary in some places of the Oceanic

data set, results in a significant over-reconstruction issue that the BC algorithm entirely avoids. Thus, for example, for the cognate set of the word 'to choose,' the human reconstruction is /piliq/. The BC algorithm reconstructs /vili/, which is a Levenshtein distance of 2 away. The CMA algorithm, however, runs into a problem. Many slightly different words such as /vili/ and /hilihilia/ are aligned together by List's algorithm for the 'to choose' cognate set. As a result, the CMA reconstruction is /fhilifinihilil/, which is a Levenshtein distance of 11 away from the human reconstruction. If all words where such mistakes occur were to be discarded from the CMA comparison (i.e. ignore all reconstructions 4 or more phonemes longer than the human ones), the average edit distance would shrink from 3.01 to 2.84 (8 words removed from the data set). The next biggest problem, I hypothesize, revolves around the *biuniqueness* criterion and context. The BC model is able to set "millions of parameters" for different contexts and the probability of sound changes in those contexts if need be, and this, while likely still an overcomplication, performs significantly better than the *biuniqueness* criterion as implemented for the CMA (for flaws of this implementation, see the discussion in Section 4.2.3.1) (Bouchard-Côté 2013, p.2). However, in the interests of space, I omit detailed context examples from the Oceanic data here. The fact that the *biuniqueness* problem arises again suggests that improving the implementation of the *biuniqueness* criterion in the CMA should be a priority.

Thus, even though currently the CMA performs worse than the Bouchard-Côté's algorithm (especially on the larger data set), the issues that create this situation (especially those revolving around *biuniqueness*) may be fixable in the future. Overall, as alluded to in Table 3.1 in Section 3.1, the CMA fulfills the *effectivenesss* heuristic worse than the BC algorithm, but the CMA performs better on both *usability* and, especially, *simplicity*. These advantages suggest that a model such as the CMA is worth the further development discussed throughout this section, even if it currently performs worse.

Furthermore, the DOCEs in the CMA program allow the user to judge the effectivenesss of the application of the CMA to a given data set, which further ameliorates the

created problems. The next section, in fact, is dedicated to discussing this next crucial aspect of the CMA — the DOCEs.

4.2.4 The DOCEs and their Quality

The goal of the DOCEs, as discussed in Chapter 1, was two-fold: to provide a practical evaluation of the reconstructions presented by the CMA and act, hypothetically, as another way of approaching the question of how 'real' a reconstruction is. In this section, I argue that the DOCES do a reasonable job of fitting both roles.

The first question, thus, is whether the DOCEs allow the user to judge the accuracy of the reconstruction. In Table 4.9, the average DOCE of each data set's reconstructions is compared with the average Levenshtein distance of the same reconstructions.

Table 4.9: Average DOCEs for each data set

Data Set	Ave. Lev. Dist.	DOCE
AHS	0.42667	0.222
PMJ	2.15	0.185
Oceanic	3.01	0.00000132

First, notably, the performance and confidence estimates line up accurately: for each pair, the high DOCE corresponds to the better score. The notable anomaly is that the comparatively high DOCE of the PMJ data set. The main reason for this high DOCE score is that an unusually high percentage of the human reconstructions seem to rely on information other than that available from the word set, as discussed in Section 4.1.1.2. This sort of mismatch occurs, as was stated earlier, quite frequently, though the small size of the Nothofer data set and the small number of languages involved increase the probability of such events. Thus, the DOCEs of smaller data sets with fewer languages are likely to have inflated DOCE scores. However, this should not impair, given that

the fact is acknowledged, the use of the DOCEs to evaluate a set of reconstructions. The most probable reason why the AHS data set does not have a too-high DOCE is that, as it is a textbook example, no other data is required to do the reconstructions. The DOCE of the Oceanic data, on the other hand, differs very significantly from both of the other two, so that its poor performance is predicted by the DOCE. In part, the average DOCE for Oceanic is so significantly lower than the other two simply because there are more internal nodes in the tree (recall that the DOCEs of all internal nodes are multiplied together and that all such scores are between 0 and 1). However, many problems arise for the CMA with such larger data sets (e.g. more missing data in a given cognate set and the *no Empty Node* issue discussed in Section 4.2.3.2), so that the correlation between larger data sets and lower DOCEs is to some degree legitimate. Thus, the user can successfully note when results are likely to be particularly poor and thus the CMA remains useful, even if this does not extend to all data sets.

Moreover, the DOCE results can be used in a much more low scale way, as there is a DOCE assigned to each proto-phoneme reconstructed by the CMA. Figure 4.7 shows the correlation between the DOCE of a proto-phoneme reconstructed for the AHS data set[6] and whether this proto-phoneme matches the proto-phoneme the human reconstructs for the same correspondence set.[7]

[6]Similar experiments were conducted for the other data sets as well, however, they are not shown and analyzed here in the interest of space. Generally, there is a correlation between the DOCE of a proto-phoneme and its correctness with the other data sets, but that correlation is weakened as many other factors (discussed above) interfere with the quality of the reconstructions.

[7]It would also be useful to consider the correlation between the quality of the reconstruction of a whole word and the DOCE of that word. However, the correlation between DOCES and the Levenshtein distances of automatic reconstructions away from the human reconstruction is even weaker than the correlation between individual reconstructed phonemes and DOCEs. It is in fact so weak as to be useless at this stage. This occurs for two reasons, neither of which are properties of the CMA. First, any weakness in the correlation between DOCEs and individual phonemes is compounded when averages over some number of phonemes and DOCEs are used, as occurs in words. Second, the expected Levenshtein distance of a word also increases with the word's length (more opportunity for wrong reconstructions), and this is in no way modeled by the average of the DOCEs of the proto-phonemes in

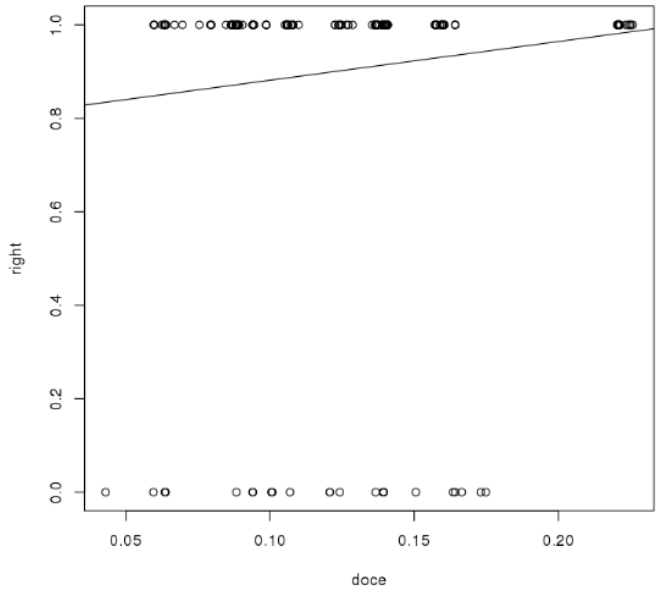

Figure 4.7: Correctness vs. DOCE of Reconstructed Proto-Phoneme

To determine the strength of the correlation between 'correctness' and DOCEs, the point-biserial correlation coefficient was calculated. As Brown 2001 points out, this type of coefficient can be effectively used to "estimate the degree of a relationship between a naturally occurring dichotomous variable [here whether the reconstructed proto-phoneme matches the human reconstruction] and an interval (or ratio) scale [here the DOCE]" (p.13). This coefficient ranges from 0 to 1, but in this case it was calculated to be 0.014. While this is strictly speaking a correlation in the right direction, it is barely noticeable. Nonetheless, I argue that this is a useful preliminary result that encourages further exploration of DOCEs as predicting the quality of the reconstruction

the word. For these two reasons, comparing the DOCEs of the reconstructions of words against their quality is not practical at the moment. Notably, however, if the quality of the CMA improves, the first reason may fall away, and the comparison (perhaps somehow adjusted for word length) may become practical again.

(in this case, in the sense of how closely a automatic reconstruction matches the human reconstruction). First, note that in this case there are only three basic types of mistakes in the data sets (the ones discussed in Section 4.2.3.1), and a bigger yet still transparent data set may yield a more definitive correlation. This is especially true since two of the mistake types are ones that do not necessarily correspond to a decrease in the DOCE. Recall from Section 4.2.3.1.1 that in the case of one of the two main mistakes (/b/ — /p/), there is an issue with the naturality matrix, and this type of problem is not reflected by the DOCE. Similarly, with the same mistake type, the DOCE score is not lowered when a mistake is made because a correspondence necessary for the *biuniqueness* criterion to come into play has not been seen yet. Thus, several of the incorrectly reconstructed phonemes (specifically, /b/) have DOCEs on the order of 0.258, higher than the average 0.224. This type of error is very likely to dissipate once the *biuniqueness* criterion is implemented differently. On the other hand, the DOCEs of the pattern three mistakes (where /ɣ/ is reconstructed instead of /w/) all show a low DOCE of 0.159. Here, the penalty from the *biuniqueness* criterion is taken into account. This suggests that even though the correlation between DOCEs is the 'correctness' of a proto-phoneme is currently low, using larger data sets and fixing the issues within the CMA with the *biuniqueness* criterion is likely to bring more useful results.

Furthermore, these results suggest that the DOCEs could be a reasonable approach to evaluating the 'reality' of the human reconstruction. In Chapter 3, the hypothesis was proposed that the DOCE of a given reconstructed proto-phoneme (and thus the average DOCE of a run) reflects the strength of the evidence (i.e. the scores along the various criteria) for reconstructing this particular proto-phoneme. If this hypothesis holds, one would expect a textbook data set, where the all the evidence should be quite strong for the sake of student linguists, to receive a significantly better DOCE than an example where even linguists have trouble agreeing on the correct answers. This hypothesis is at least partially supported by the AHS and Oceanic data sets. The ABVD provides two sets of reconstructions (one by Blust and the other by for the Oceanic data sets, and

there is an average Levenshtein distance of 1.06 between their reconstructions, across all words that both reconstructed (data taken from p.12-13 of BC appendix). Thus, the hypothesis would suggest that the AHS data set would receive a significantly higher DOCE than the Oceanic data set, where the possibly arguments using the data are not as strong. This is confirmed here. Thus, tentatively, the hypothesis that DOCEs reflect the strength of the arguments to be advanced for a reconstruction (and thus, as stated in the Introduction, the reality of said reconstruction) is supported by the data available.

The next step is to connect these results with the argument concerning the 'reality of reconstruction,' which DOCEs were originally intended (see Section 1.2) to shed light upon. As discussed in Section 1.1, DOCEs represent the strength of the evidence for the produced reconstruction, and the strength of such evidence is an interesting way of looking at the question of the reality of reconstructions. Thus, since the tests suggest that DOCEs accurately represent this evidence, they can be used as an empirical evaluation of the 'reality' of given reconstructions. Thus, the analysis would suggest that people are more likely to once have used the exact forms reconstructed for PMJ than the exact forms reconstructed for Oceanic.

Notably, the small number of data sets used here is not sufficient to entirely support the two hypotheses proposed above: that DOCEs reflect the quality of the reconstruction and that DOCEs reflect the 'reality' of said reconstruction. However, it is significant that with the little data available both of these hypotheses are supported.

Overall, thus, the results of testing the CMA algorithm against three data sets, suggests that the CMA algorithm is worth developing further given its results, and that the available data supports the hypothesis that DOCEs as defined for the CMA can reflect both the quality of the reconstruction and the 'reality' of it.

Chapter 5

Conclusion

The Comparative Method Algorithm presented here implements a version of the comparative method, producing proto-phonemes, proto-words, and DOCEs for a proto-language, given an input of sets of words with the same meaning from some number of related languages. For the 'cognates and correspondences' step of the algorithm, the algorithm of List 2012 was used to find cognate sets and identify correspondences between them. Using this information, the CMA then performs the rest of the analysis and produces the output.

Overall, this algorithm succeeded in the goals set out in the Chapter 1. First, it fits the four heuristics set out in Section 1.2, as illustrated in Table 3.1. First, the model used is fairly **simple**, at least in comparison to the vastly more complex Bouchard-Côté model. If the algorithm can be improved further to entirely match the performance of this better developed algorithm (e.g. if the *biuniqueness* implementation could be fixed), this simplicity will be a significant advantage. Second, the CMA fits the heuristic of **usability** well, as it as adaptable to a user linguist's linguistic theory in a way neither of the other existing algorithms are: the user can both set their own naturality matrix and determine the weights of the various criteria that are involved in evaluating a reconstruction. Furthermore, the CMA fulfills the heuristic of **theoretic soundness** by using the same types of evidence as the human linguist: reconstructing proto-phonemes from correspondences rather than whole words, using complex phylogenetic trees for the analysis, and using the criteria for evaluating candidate proto-phonemes that are considered most important in the literature.

Finally, the CMA meets, to some significant degree, the **effectiveness** goals that were set out in the introduction. One of the main goals of the CMA was to expedite the work of the human linguist, for which the reconstructions produced by the CMA had to be as close as possible to those produced by the human. In it's current state the CMA performs better along this sub-heuristic than Oakes but somewhat worse than the Bouchard-Côté et al. algorithm. However, the fact that this much less developed algorithm produces comparably to the results of the long term efforts of Bouchard-Côté et al. strong suggests that the CMA model is a promising beginning that must be developed further. As stated in Section 1.2, the difference between the CMA and Bouchard-Côté et al.'s algorithm is a way of looking at the dichotomy between statistical and theory-based approaches to algorithmic reconstruction, and thus further exploration of the promising theoretic approach is especially interesting.

The second goal of the CMA was to provide DOCEs that would serve as a way of estimating both the quality of the reconstructions output by the algorithm and the 'reality' of reconstructions. For the preliminary test presented in the results section (Chapter 4), both of these goals were accomplished: there is a correlation between the DOCEs of tuples and whole runs of the algorithm and their respective DOCEs, and the Oceanic data set that offered more ambiguous evidence and would be expected to produce reconstructions less likely to directly correlate with forms pronounced by it's speakers — did in fact receive lower DOCEs. Thus, the preliminary results suggest that the DOCEs are effective in fulfilling their roles, though further testing would be useful.

All of the above results suggests the CMA should be both tested, explored, developed further, and there are a variety of forms such testing and development could take. In terms of testing, the most interesting venue is finding further data sets where more than one linguist has offered a reconstruction, and comparing the DOCEs of the reconstruction from that data set with the average Levenshtein differences between the reconstructions of the linguists. If lower DOCEs correlate with higher distances, that would be strong evidence that the DOCEs are in fact useful in providing an empirical

approach to the question of the reality of reconstructions.

Yet, even though the CMA already offers a wide variety of uses for historical linguists, there are many ways in which the algorithm could be further developed. First, as discussed throughout Chapter 4, the implementation of the *biuniqueness* criterion has to be changed. Another interesting modification would be to include more criteria, such as those described in Section 3.1.2, and thus improve further the **usability** of the model. A final interesting idea would be to integrate the statistical and theoretic approaches, so that the CMA would start out with a theory-based naturality matrix and then allow the matrix to evolve to better fit the data using the same sorts of statistical tools that Bouchard-Côté et al. use. There are a plethora of such different ways to improve the CMA, and each of them is likely to significantly raise the quality.

In conclusion, the CMA already offers a rich model that allows for new and useful types of analysis, and it can be easily improved further suggesting that further work on the algorithm would be beneficial.

Bibliography

Abney, S. (2011), 'Data-intensive experimental linguistics', *Linguistic Issues in Language Technology* .

Blevins, J. (2008), Naturalness and iconicity in language, *in* K. Willems and L. D. Cuypere, eds, 'Natural and Unnatural Sound Patterns: A Pocket Field Guide', John Benjamins, Amsterdam, pp. 121–148.

Blust, R. (1993), 'Central and central-eastern malayo-polynesian', *Oceanic Linguistics* **32**(2).

Bouchard-Côté, A., Griffiths, T. and Klein, D. (2008), A probabilistic approach to language, *in* 'Proceedings of NIPS'.

Bouchard-Côté, A., Griffiths, T. L. and Klein, D. (2009), Improved reconstruction of protolanguage word forms, *in* 'Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics'.

Bouchard-Côté, A., Hall, D., Griffiths, T. L. and Klein, D. (2013*a*), 'Automated reconstruction of ancient languages using probabilistic models of sound change', *Proceedings of the National Academy of Sciences* .

Bouchard-Côté, A., Hall, D., Griffiths, T. L. and Klein, D. (2013*b*), 'Supplement to "automated reconstruction of ancient languages using probabilistic models of sound change"', *Proceedings of the National Academy of Sciences* .

Bouchard-Côté, A., Liang, P., Griffiths, T. L. and Klein, D. (2007), A probabilistic approach to diachronic phonology, *in* 'Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning', pp. 887–896.

Bowern, C. (n.d.), Solutions sets for homework assignment two. Solution Sets to Textbook Example.

Brown, C. H., Holman, E. W. and Wichmann, S. (2011), 'Sound correspondences in the world's languages'.

Brown, J. D. (2001), 'Point-biserial correlation coefficients', *LT Testing and Evlution SIG Newsletter* **5**(3), 13–17.

Campbell, L. (1998), *Historical Linguistics*, Edinburgh University Press.

Crowley, T. (1992), *An Introduction to Historical Linguistics*, Oxford University Press.

Crowley, T. and Bowern, C. (2010), *An Introduction to Historical Linguistics*, 4 edn, Oxford University Press.

Delmestri, A. (2011), Data Driven Models for Language Evolution, PhD thesis, University of Trento.

Eddington, D. (2008), 'Linguistics and the scientific method', *Southwest Journal of Linguistics* **27**, 1–16.

Fox, A. (1995), *Linguistic Reconstruction: An Introduction to Theory and Method*, Oxford University Press.

Gilman, S. (2012), 'Comparative method algorithm', *Cambridge Occasional Papers in Linguistics* **6**(5), 131–175.

Greenhill, S.J., B. R. and Gray, R. (2008), 'The austronesian basic vocabulary database: From bioinformatics to lexomics', *Evolutionary Biology* **4**, 271–283.

Hamed, B. M. and Flavier, S. (2009), Unidia: A database for deriving diachronic universals, *in* M. Dufresne, F. Dupuis and E. Vocaj, eds, 'Historical Linguistics 2007: Selected papers from the 18th International Conference on Historical Linguistics', Current Issues in Linguistic Theory, John Benjamins, Amsterdam, pp. 259–268.

Hamed, B. M. and Flavier, S. (2011), 'The diadm project: A web-based platform for diachronic data and models'.
URL: *http://www.diadm.ish-lyon.cnrs.fr/*

Hauer, B. and Kondrak, G. (2011), Clustering semantically equivalent words into cognate sets in multilingual lists, *in* 'The 5th International Joint Conference on Natural Language Processing'.

Johnson, M. (2011), 'How relevant is linguistics to computational linguistics?', *Linguistic Issues in Language Technology* **12**.

Kessler, B. (2005), 'Phonetic comparison algorithms', *Transactions of the Philological Society* **103**(2), 243–260.

Lass, R. (1993), How real(ist) are reconstructions, *in* C. Jones, ed., 'Historical Linguistics: Problems and Perspectives', Longman, pp. 156–189.

List, J.-M. (2012*a*), Lexstat: Automatic detection of cognates in multilingual wordlists, *in* 'Proceedings of EACL, Joint Workshop of LINGVIS and UNCLH', pp. 117–125.

List, J.-M. (2012*b*), 'Sca: Phonetic alignment based on sound classes', *New directions in logic, language, and computation* pp. 32–51.

Lynch, J. (n.d.), 'The indo-european language family tree'.

Nichols, J. (1995), Diachronically stable structural features, *in* H. Andersen, ed., 'Historical Linguistics 1993: Papers from the Eleventh International Conference on Historical Linguistics', Amsterdam: John Benjamins, pp. 337–356.

Nothofer, B. (1975), *The Reconstruction of Proto-Malayo-Javanic*, 'S-Gravenhage - Martinus Nijhoff.

Oakes, M. P. (2000), 'Computer estimation of vocabulary in a protolanguage from word lists in four daughter languages', *Journal of Quantitative Linguistics* **7**(3).

Olle, E., Petur, H. and Mikael, P. (2008), The beginnings of a database for historical sound change, *in* 'Papers from the 21st Swedish Phonetics Conference', pp. 101–104.

Pulgram, E. (1959), 'Proto-indo-european reality and reconstruction', *Language* **35**(3), 421–6.

Ringe, D. (1992), 'On calculating the factor of chance in language comparison', *Transactions of the American Philosophical Society* .

Sokal, R. and Michener, C. (1958), 'A statistical method for evaluating systematic relationships', *Univ. Kansas Sci. Bull.* **38**, 1409–1438.

Steiner, L., Stadler, P. F. and Cysouw, M. (2011), 'A pipeline for computational historical linguistics', *Language Dynamics and Change* **1**(1), 89–127.

Trask, R. (1996), *Historical Linguistics*, 2 edn, Oxford University Press.

Trask, R., ed. (2000), *The dictionary of historical and comparative linguistics*, Edinburgh University Press.

Trask, R. and Millar, R. M. (2007), *Historical Linguistics*, Hodder Education.

Warnow, T., Evans, S., Ringe, D. and Nakleh, L. (75–), A stochastic model of language evolution that incorporates homoplasy and borrowing, *in* 'Phylogenetic Methods and Preshistory of Languages', chapter 7.

Wurzel, W. U. (1989), *Inflectional Morphology and Naturalness*, Akademie-Verlag Berlin.