Sonority and Reduplication in Nakanai and Nuxalk (Bella Coola)

Katy Carlson

1. Introduction

Recent work on reduplication has focused on deriving the segmental content and shape of reduplicants from general, non-stipulative phonological constraints. This paper will argue that the reduplicative patterns of Nakanai (Austronesian) and Nuxalk (Salish) are surprisingly similar, in that the variety of partial reduplication patterns shown by both languages can be derived using simple markedness constraints on sonority. In Nakanai (Johnston 1980) reduplication, sonority constraints determine which vowels are copied. In Nuxalk (Bagemihl 1991, Newman 1971, Nater 1984) reduplication, sonority constraints determine the position of the reduplicant. The intriguing differences between the two languages' reduplication patterns stem from different rankings of sonority markedness constraints with respect to Base-Reduplicant Identity constraints; such ranking differences are predicted in Optimality Theory, and form a typological argument for the constraints themselves.

Sonority has long been important in syllable structure, with many pointing out that syllables have a preferred sonority pattern (cf. Clements 1990, Zec 1995 for recent work); a syllable with this pattern can be said to be unmarked. One hallmark of the unmarked syllable shape is that there is a sharp rise in sonority between the onset and the nucleus; thus a very non-sonorant onset and a very sonorous nucleus are preferred. This is an intralinguistic preference as well as a crosslinguistic preference, since few languages allow voiceless obstruents to be nuclei (Imdlawn Tashlhiyt Berber, for example, is rather a rarity in this respect (Dell and Elmedlaoui 1985)). Nakanai reduplication shows a preference for low vowel peaks over high vowels, copying the low (e.g., *tatuga*, not **tutuga*). Nuxalk shows a preference for sonorant consonant and vowel peaks over obstruent peaks, infixing the reduplicant in order to select a sonorant nucleus (e.g., *sqmqma-i*, not **sqsqma-i*). Another sonority effect involves diphthongs. Diphthongs that rise in sonority, such as *ia*, do not follow the ideal sonority slope for a syllable, as they slow down the rise of sonority between the onset and nucleus (Rosenthall 1994); in Nakanai, these diphthongs are excluded from reduplicants (e.g., *paipati* from *pati*, vs. **piapita* from *pita*). This paper argues that the reduplicative patterns above follow from a desire to have unmarked syllables in reduplication.

This paper will be structured as follows. Section 2 will introduce the general Optimality Theory framework of the paper, and some of the important constraints that will play a role in the analyses. Section 3 will outline the Nakanai facts and give an analysis, section 4 will cover the Nuxalk facts and analysis, and the paper will conclude with section 5.[1]

[1] Independent research by Philip Spaelti (1997) has led to a similar analysis of Nakanai reduplication in the context of his dissertation on languages with multiple reduplication patterns.

2. Reduplication in Optimality Theory

This paper will use the framework of Optimality Theory (Prince and Smolensky 1993), in which candidate outputs are created by GEN from each input, and whichever candidate best satisfies the constraints of the language is the selected output. Constraints are universal, and ranked and violable; they are violated minimally, under duress from higher-ranked constraints. There are two major constraint types: constraints regulating the faithfulness of the output to the input, and constraints on the markedness of the output. The faithfulness constraints used here are the Correspondence Theory constraints (McCarthy and Prince 1995); these gauge violations by setting up a correspondence relation between two strings, and then comparing correspondent elements in the strings. A fully faithful output, for example, contains identical correspondents of each element in the input.

The sonority restrictions on syllable structure mentioned above are visibly active in the syllabification systems of many languages, as well as the reduplicative systems discussed here. So they can be expressed as markedness constraints that are part of UG. In particular, the preference for sonorous nuclei is captured by the Peak hierarchy of constraints (Prince and Smolensky 1993). These rank all segments, in order of sonority, as to their fitness to be syllable nuclei:

(1) *P/t,k >> ... >> *P/l >> ... >> *P/i,u >> *P/e,o >> *P/a

According to this hierarchy, no language should prefer to have a /t/ in the nucleus over a vowel, and most languages probably won't allow /t/ to be a nucleus at all; /l/ is better, since it is a sonorant consonant, and the low vowels are best of all. This accords with intuitions about normal syllable structures across languages. The preference for a steep initial sonority slope can be partially captured by the following constraint on diphthongs (after Rosenthall 1994):

(2) *LHDip: Diphthongs must not rise in sonority.

Now, these markedness constraints are not the only constraints in the rankings for these languages. First of all, we need to make sure that their effects are restricted enough. In Nakanai and Nuxalk, as in many other languages, reduplicants are less marked in particular ways than the rest of the outputs in the language. That is, some constraints appear to be active only in the reduplicative domain. This is captured by the 'emergence of the unmarked' ranking (McCarthy and Prince 1994, 1995). Using this ranking schema, the faithfulness constraints for the language in general (the Input-Output (IO) mapping) have to outrank the sonority markedness constraints, so that high vowels and rising-sonority diphthongs are allowed (e.g., Nakanai has roots like *ligi* and *sio*). But the markedness constraints have to outrank at least one reduplicant faithfulness constraint, so that their effects are seen in reduplication (e.g., *sio* cannot fully reduplicate as *_siosio_):

(3) IO-Faith >> Markedness constraint(s) >> some Base-Reduplicant (BR) Faith constraint

This general ranking outline will be seen in both languages. A simple faithfulness constraint that will be ranked below markedness constraints in both systems is Max, specifically Max-BR, which penalizes non-reduplication of segments in the base:

(4) Max-BR: Every segment in the base has a correspondent in the reduplicant

Many of the faithfulness constraints active in the the Nakanai and Nuxalk reduplication systems are the same; however, they are ranked differently with respect to the sonority constraints, creating the differences between the languages. Some of the important constraints can be formulated as in (5):

(5) ALIGN (RED, R, PWd, L): Align the right edge of the reduplicant with the left edge of a prosodic word (= the reduplicant is a prefix)
L-ANCHOR: The reduplicant begins with the correspondent of the base-initial segment
CONTIGUITY: The segments in the reduplicant are contiguous in the base

The first of these is a familiar constraint that uses Alignment to state the normal position of an affix, such as a prefixing reduplicant, with respect to a prosodic category (the Prosodic Word here). L-ANCHOR and CONTIGUITY are Correspondence constraints. McCarthy and Prince (1995) noted that reduplicants usually depend on one edge of the base and reduplicate the edgemost segment; prefixing reduplication systems are often ones in which copying begins with the leftmost segment of the word (and conversely for suffixing reduplication and the rightmost segment). This phenomenon is known as Anchoring. Note that the Anchoring constraint refers to the base, rather than an independent prosodic category like the word. This allows the Anchoring domain to be defined by the position of the reduplicant; for a prefixal reduplicant, the base is defined as the string that follows the reduplicant.[2] Thus Anchoring and Alignment violations are quite separable. Finally, CONTIGUITY is a constraint against skipping segments of the base.

These constraints can be interleaved with the markedness constraints above to create a nice typology. In Nakanai, L-ANCHOR and Alignment are ranked high, while CONTIGUITY is low-ranked. This means that the first segment of the base is always reduplicated, and the reduplicant is always a prefix, but segments can be skipped, as in *papita*. In Nuxalk, on the other hand, L-ANCHOR and CONTIGUITY are high, while Alignment is lower-ranked. Thus the first segment of the base is always reduplicated, and segments of the base can’t be skipped, but the reduplicant can become an infix under duress from markedness constraints: *st’q^{w}lq^{w}lus*. Pursuing this typology a little bit further, a form of Sanskrit reduplication has Alignment high, but L-ANCHOR low-ranked; the reduplicant is always a prefix, but the first segment of the base is not always reduplicated: *tustu*. The complete typology of these constraints has not been explored, but it is striking how these languages with such similar markedness restrictions fit neatly into one. The analyses for each language will discuss the particular rankings necessary for their reduplicative systems.

Finally, since a reduplicant does not have fixed segmental content, the system must decide whether it reduplicates all of a word, no matter what size it is, or whether the reduplicant specifically targets only a sub-constituent of a word. Work in the Generalized Template theory has focused on deriving reduplicant shapes from general markedness constraints (e.g., McCarthy and Prince 1994, Gafos 1996, Spaelti 1997, Urbanczyk 1995, 1996; related to work by Shaw 1987 and Steriade 1988). This strategy is very successful with patterns where the shape of the reduplicant depends on the segmental content of the base, for example, as in some of the Nakanai patterns; however, there are also patterns where some pre-specification is necessary. This type of lexical effect will be captured here using a PROSTARG(X) constraint (Gafos 1997), which states that the reduplicant should be the size of a particular prosodic constituent X; X can vary, for any

[2] In Correspondence Theory, the base is always defined with respect to the reduplicant; see McCarthy and Prince 1993 and Urbanczyk 1996 for some discussion.

given language or reduplicative pattern. The prosodic target X can also be set to nothing: PT(0), which means that the PT constraint is vacuously satisfied by any reduplicant shape, and the markedness constraints of the language derive the size. This parametrized constraint is attractive because it can be fixed in the constraint ranking for a language and pattern, and still derive different-sized reduplicants. In Nakanai, some roots take PROSTARG(0), which means that there is no predetermined reduplicant size; other roots take PROSTARG(Syllable), which fixes the reduplicant size at a syllable. In Nuxalk, it appears that all reduplication patterns are maximally syllable-sized, so PROSTARG(Syllable) is active for all of them.

3. Nakanai

In this section I will introduce Nakanai, and discuss how sonority constraints affect the Nakanai reduplicative affix, and which sonority constraints are seen to be active. The predictive ability of this system will also be explored. The vowel patterns in roots are the major determinants of reduplicant size and shape. And in Nakanai, unlike Nuxalk, reduplication satisfies markedness constraints by failing to reduplicate segments or skipping segments (MAX-BR and CONTIGUITY violations), while Alignment and L-Anchoring are fully satisfied.

3.1. About Nakanai

Nakanai is an Austronesian language of the West New Britain province of Papua New Guinea. The data in this paper come from Johnston (1980), who focused his fieldwork on the Bileki dialect of Nakanai. This dialect is the most prestigious and has the greatest number of speakers, including many of the speakers of other dialects.

The phoneme inventory of Bileki Nakanai is as follows:

(6) Phonemes

Consonants	Labial	Coronal	Dorsal	Glottal
Stops	p, b	t, d	k, g	
Fricatives	β	s		h
Nasals	m	(n)	(ŋ)	
Liquids		l, r		

Vowels	Front	Central	Back
High	i		u
Mid	e		o
Low		a	

The coronal and dorsal nasals are found only in borrowed words, and are not part of the core phoneme inventory of the language. The other four dialects of Nakanai lack the phoneme /h/, which may relate to the fact that *h*-initial words in Bileki Nakanai show some irregularities in reduplication, as we will see later.

There are no closed syllables in Nakanai, except in some borrowed words, and onsets are not necessary. The attested syllable structures are thus V(V) and CV(V), with diphthongs and long vowels allowed. Stress falls on the penultimate mora of the word, and so I assume that the last two moras form a metrical foot.[3] With the exception of a few dependent function words, all words are at least bimoraic. In this characterization of the Nakanai syllabic structure I depart from Johnston (1980), who claims instead that diphthongs and long vowels are not allowed, so that each vowel forms the nucleus of its own syllable. He presents little evidence for this conclusion, however, and a diphthongal analysis provides a better understanding of the interesting vowel pattern effects that will be presented below.

3.1.1. Purposes of Reduplication

Reduplication is used in Nakanai for several purposes. Agreement with a non-singular clause topic is marked on the manner adverb or verb of the clause by reduplication, although this agreement is optional in most circumstances. Reduplication also appears on a verb to mark the continuative or habituative aspect, as well as to derive intransitive verbs from transitive ones. And in conjunction with the perfective suffix, it marks the imperfective aspect of the verb. On nouns, reduplication forms the collective plural, and makes abstract noun stems into concrete nouns. It is also used to derive distributive adverbs from numerals. Finally, there are numerous lexically-reduplicated words in the language, for which no unreduplicated stem is found, but which obey the usual rules for reduplication. There are several patterns of reduplication in Nakanai, but reduplication does not vary by function; that is, the factors determining which form reduplication will take for each word do not include the function of reduplication in that word. Johnston (1980) states that the reduplication patterns are phonologically conditioned, which is true up to a point. There are definite generalizations active in determining the form of reduplicants, although the patterns are only partially predictable.

3.1.2. Patterns of Reduplication

Reduplication takes the bimoraic final foot of the word as its base. Pattern 1 involves complete reduplication of the base foot, and patterns 2-5 involve reduplication of part of the base.

(7) Reduplicative Patterns 1-5 (reduplicants are underlined)

1. $C_1V_1C_2V_2$

ligi	ligiligi	‘hurting’
palo	palopalo	‘wakening/baskets’
kuruve	kuruveruve	‘many sweet potatoes’

[3] I make no claims about the possible footing of the rest of the word, in words longer than two moras. Only this final foot is used in my analysis.

2. C_1V_1		
buli	bubuli	‘rolling’
kapu	kakapu	‘pulping’
giu	gigiu	‘peeling’
3. C_1V_2		
tuga	tatuga	‘walking’
sile	sesile	‘tearing’
sio	sosio	‘carrying on ceremonial litter’
4. $C_1V_1V_2$		
pati	paipati	‘floating’
gove	goegove	‘mountains’
bau	baubau	‘singing’
5. V_1C_1		
abi	ababi	‘getting’
oli	ololi	‘digging’
haro	hararo	‘days’
kaiamo	kaiamamo	‘residents of Kaiamo village’

Most of the examples shown above are with bimoraic words, which are very common in Nakanai. In these words the reduplicative affix looks like a normal prefix, with the exception of the *h*-initial form. However, in the longer words (such as *kuruve*) it is clear that the reduplication is not actually prefixed to the word, but appears before the bimoraic metrical foot of the word (i.e. the reduplicative base). Thus the reduplicant is actually prefixed to the foot.[4]

There are important generalizations about the distribution and form of the reduplicative patterns. The vowel patterns in roots are most important in determining which reduplicative pattern they will take, but here I will sketch what we learn from the consonantal patterns. Since the reduplicative base is the foot of the root, we will look only at the content of the foot, not the whole root. The following table shows the five main types of bases:

(8) Reduplicative patterns by base type

Base	Pattern	Examples
two obstruents	2, 3, 4	kakapu, tatuga, paipati
1-2 sonorants	1, 2, 3, 4	ligiligi, tataro, sesile, taitari
V-initial	1, 5	osaosa, ababi
CVV	1=4, 2, 3	baubau, sasae, lalea
h-stem	5	hararo

Bases with two obstruent consonants, such as *kapu*, can reduplicate by patterns 2, 3, or 4, the partial reduplicative patterns. Bases containing one or more sonorant consonants, such as *ligi*, can take any of the four consonant-initial reduplication patterns. This difference in the distribution of

[4] This is not precisely true, since the foot of a form like *ababi*, from the V-initial root *abi*, has clearly become (*babi*); this complication will be dealt with in section 3.5.1.

reduplication patterns, which is mediated by the consonants, will be discussed in section 3.4.3. Vowel-initial bases can take pattern 5 (VC), and fail to reduplicate the second vowel, or they can reduplicate completely (VCV). CVV bases such as *bau* can reduplicate completely, which by definition is equivalent in shape to pattern 4 (CVV) reduplication; or they can reduplicate with one or the other vowel left out. There are only a few consonant-initial bases like *haro* that take pattern 5, and several of them begin with *h*. Also, no *h*-stems take any other patterns of reduplication. So the *h*-stems and the other few consonantal roots that take pattern 5 are somewhat exceptional, in that they reduplicate as if they were vowel-initial. We return to this fact in section 3.5.1. Now we turn to the really predictive root patterns.

3.2. Vowel-Related Patterns and Constraints

This section will discuss the vowel patterns found in Nakanai bases and reduplicants. The generalizations that are seen in the vowel-pattern tables will motivate the substantive sonority constraints that were introduced in section 2.

3.2.1. Vowel Patterns

All of the data surveyed here are from Johnston (1980), which is not an exhaustive data set; however, I believe the patterns that emerge are nonetheless significant and form true generalizations. There are few obvious regularities in the vowel pairs of bases that take reduplicative pattern 1 (CVCV), so the table is not shown here.

(9) Number of roots per pattern, by vowel pairs
Pattern 3: CV_2
Generalization: V_2 more sonorous than V_1: more sonorous vowel is reduplicated

3:V_1 / V_2	i	e	a	o	u
i		2	2	2	
e			4		
a					
o			2		
u			4		

Pattern 4: CV_1V_2
Generalization: V_1 more sonorous than V_2 (or equally sonorous): falling-sonority diphthong is reduplicated

4:V_1 / V_2	i	e	a	o	u
i					
e	2				
a	5	1		1	4
o	1	2			3
u					

The tables above for patterns 3 and 4 are striking, since the vowel pairs are completely complementary. Bases that take pattern 3 (CV_2), in which only the second vowel is reduplicated, have vowel pairs that rise in sonority, such as *i* and *e*, or *o* and *a*. Bases taking pattern 4 (CVV), the only pattern in which the two vowels are reduplicated as a diphthong, have vowel pairs that fall in sonority or stay level, such as *a* and *i*, or *o* and *e*. I claim that this asymmetry shows a restriction on what diphthongs are allowed in reduplicants: *ai* vs. **ia*, where the falling-sonority diphthong is allowed and the rising-sonority diphthong is not. Thus a root like *pati* can reduplicate a diphthong in *paipati*, but a root like *pita* could not reduplicate the diphthong in **piapita*. I will show that alternate syllabifications which avoid diphthongs are ruled out by the Onset constraint. Bases with identical vowels are considered to be taking pattern 2 (CV_1), when only one of the vowels is reduplicated.[5] Interestingly, they never take pattern 4 (CVV), so no long vowels are created in reduplicants, either.

Pattern 5: VC
Generalization: V_1 more sonorous than V_2 (or equally sonorous): first vowel is reduplicated

5:V_1 / V_2	i	e	a	o	u
i	1			1	
e				1	
a	4		2	4	3
o	3				
u					

The bases listed in the table for pattern 5 (VC) contain mostly identical or falling sonority vowel pairs. This regularity could be driven by requirements on the diphthongs created by reduplication, as above. Vowel-initial roots which reduplicate entirely will create diphthongs between the final reduplicated vowel and the initial base vowel: *os**ao**sa* (again, Onset will keep the vowels from being syllabified separately). If the vowels fall in sonority through the root, as in *abi*,

[5] Obviously they could be considered as CV_2 reduplication instead, since one can't tell which vowel is reduplicated when the vowels are identical.

then complete reduplication would create a less-preferred rising-sonority diphthong at the Reduplicant-Base juncture: **ab**i**a**bi*, as opposed to *ababi*.

Pattern 2: CV$_1$
Generalization: V$_1$ more sonorous than V$_2$ (or equally sonorous): first vowel is reduplicated

2:V$_1$ / V$_2$	i	e	a	o	u
i	6	1	3		2
e	2	7	1	2	2
a	4	2	8	3	4
o	4		1	5	6
u	5				6

Finally, among the 74 roots taking reduplicative pattern 2 (CV$_1$), only 6 of them show vowel pairs of rising sonority. That means that this reduplication copies the higher (i.e. lower sonority) vowel of the root only 6 times; the rest of the time it copies the lower of the two (if they are not identical). This correlates with the fact that the roots taking pattern 3 (CV$_2$) have vowel pairs that rise in sonority, so that the lower (i.e. higher sonority) vowel is always copied in this pattern also. And in pattern 4 (CVV), the allowable diphthongs are those in which the lower vowel is first and forms the peak of the syllable. To sum up, these tables show several important generalizations about Nakanai reduplicative patterns:

a. lower vowels are favored as peaks in patterns 2, 3, 4
b. reduplication can not create long vowels
c. reduplication can not create rising-sonority diphthongs

3.3. Vowel-Related Markedness Constraints and Rankings for Nakanai

Now we have seen evidence for the sonority constraints introduced in section 2. In Nakanai, we only need to see the portion of the Peak constraint hierarchy that ranks vowels (the constraints against consonant Peaks are undominated in the language as a whole). The relevant constraints favor low vowels first, then mid vowels, and then high vowels:

(10) Peak Constraints: *P/i,u >> *P/e,o >> *P/a

These Peak constraints are ranked below Input-Output Faithfulness, since many roots and even reduplicants in Nakanai contain high vowel peaks, but they play a role in deciding which vowels in bases will be reduplicated. For example, they will choose a form like *papita* over **pipita*.

All languages do allow short vowels to be syllable nuclei. But languages (and reduplication systems) can vary as to whether they allow diphthongs and long vowels, and what types they allow (see Rosenthall 1994, Kaye 1983). Constraints on diphthongs were seen to be active in Nakanai reduplication. We begin with a general markedness constraint against diphthongs:

(11) *DIPH: Diphthongs are prohibited

This is a very low-ranked constraint, so it is only seen to be active in a few tableaux. Then there is the more specific constraint on the form of diphthongs, *LHDIP, which prohibits rising-sonority diphthongs.[6]

(12) *LHDIP: Diphthongs must not rise in sonority

Due to its ranking between IO-Faith and MAX-BR, this constraint will only emerge in the reduplicative domain, to produce the preferred syllable-initial sonority slope. And finally, another vocalic constraint that we see active is one against long vowels:

(13) NOLONGVOWEL: Long vowels are prohibited (Rosenthall 1994)

This constraint is ranked under IO-Faith, since roots of the language can contain long vowels, but above MAX-BR, since no reduplicants contain long vowels.

In general, Nakanai reduplication copies all or part of the last two moras of roots, so the final foot is the base of reduplication (recall the patterns shown in (7)). This is captured by the following Alignment constraint: ALIGN (RED, R, Foot, L). This constraint ensures that the reduplicative affix is prefixed to the foot of the root, by compelling the right edge of the reduplicant to align with the left edge of the foot. The base is never affected by reduplication, besides having the reduplicative morpheme added. This shows that IO Faithfulness constraints are highly ranked. And reduplicants only differ from their bases by deletion; that is, reduplicants are always the same size or smaller than their bases, with no foreign segments added and no featural mismatches between correspondents. That shows that MAX-BR is low-ranked, while other B-R faithfulness constraints are highly ranked, such as DEP-BR (no epenthesis) and IDENT constraints. Also, every reduplicant begins with the correspondent of the base-initial segment, which is captured by the Correspondence constraint L-ANCHOR.

Finally, several other constraints mentioned in section 2 are active in the Nakanai reduplicative system. One is the Correspondence constraint CONTIGUITY, which is violated in forms where segments of the base are skipped over in the reduplicant, such as *papita*. Also playing a role are the syllabic markedness constraints ONSET and NOCODA. ONSET favors syllables with onsets; this constraint is ranked under IO-Faith, since the language allows onsetless syllables, but over MAX-BR since it is active in reduplication. NOCODA is obeyed in the language as a whole, so it is ranked quite high.

3.4. Analysis in Detail

With the constraints discussed above, the entire set of reduplicative patterns of Nakanai can be generated. This could be a hollow claim, since that statement could be true in a case where each separate pattern needed an entirely separate ranking. Or it could be a very strong claim, such that one ranking would explain all the patterns of all the roots. In fact, it is not possible to fully predict what reduplicative pattern each Nakanai root will take, due to contrasts such as *kakapu* (pattern 2) but *gaugapu* (pattern 4); this pair shows some of the lexical unpredictability of the patterns. But the patterns can actually be generated using only one major constraint ranking, and a re-ranking of only one faithfulness constraint.The first ranking of these constraints will generate the majority of the observed reduplicative patterns, and the second ranking produces pattern 2 (CV_1).

[6] This is very similar in effect to SonFall, a constraint from Rosenthall 1994. He also uses the fact that rising-sonority diphthongs often pattern as if they are monomoraic; this distinction will not be explored here, since such diphthongs are the ones prohibited in reduplication.

3.4.1. Ranking Demonstrations

I will begin with a few small tableaux, to show some of the important constraint rankings. This tableau shows two of the markedness constraints ranked over Max-BR.

(14) *LHDIP, ONSET >> MAX-BR

/RED+pita/	*LH-DIP	ONSET	MAX-BR
a. pia.pi.ta	*!		*
b. pi.a.pita		*!	*
☞c. pa.pi.ta			**

This root contains two obstruents, which will never both be reduplicated (see the discussion in section 3.4.3). Thus the candidates shown reduplicate the first consonant and the vowels. Candidate (a), which reduplicates both vowels, fails on *LHDIP because there is a rising-sonority diphthong in the reduplicant. Candidate (b), which tries to avoid such a violation by syllabifying the vowels separately, fails because ONSET is ranked high enough to penalize candidates with extra onsetless syllables. Thus candidate (c), which avoids a *LHDIP violation by failing to reduplicate the first vowel, is the winner (pattern 4). Another possible candidate would be *_paipita_, with a preferred falling-sonority diphthong; this fails because metathesis is ruled out by high-ranking Linearity (McCarthy and Prince 1995). And returning to candidate (b), illicit vowel patterns are never saved from deletion by separate syllabification in Nakanai reduplication; most of the tableaux in the following section will not show candidates with extraneous ONSET violations, for this reason.[7]

The following tableau shows the vowel-markedness constraints which ensure that candidate (b) above wins over another likely alternative, *_pipita_.

(15) MAX-BR >> *P/i,u >> CONTIG

/RED+pita/	MAX-BR	*P/i,u	CONTIG
a. pi.pi.ta	**	*!	
☞b. pa.pi.ta	**		**

Candidate (b) wins in (15) because candidate (a) contains a high vowel peak in the reduplicant. This is worse than the second candidate's skipping of two segments in the base to form the reduplicant, a double CONTIGUITY violation.[8] The tableau above showed that CONTIGUITY is dominated by *P/i,u, and the next one shows that CONTIGUITY is also dominated by *P/e,o.

[7] Another explanation for this fact would be that the Peak constraints are ranked high enough to rule out having another high or mid-vowel peak; thus _pi.a.pita_ would add an extra violation of *P/i,u, and that would cause it to be dispreferred. This explanation would not work for all cases; besides, the ONSET constraint is separately useful .

[8] One can see that both of the candidates in (15), and all output candidates which are faithful to the base, contain another high vowel peak (in the base _pita_), but the marks for that peak will cancel each other out and thus will not be shown in tableaux to follow. Any candidate which eliminates the high vowel in the base will violate undominated IO-Faith and lose, anyway. Peak violations will be assessed for each vowel in a diphthong; violations of the lowest-ranked Peak constraint (*P/a) are never fatal, though, and will not be seen.

(16) *P/e,o >> CONTIG, *P/a

/RED+kea/	*P/e,o	CONTIG	*P/a
a. ke.kea	*!		
☞b. ka.kea		*	*

Since the vowel pattern in this root is a rising-sonority diphthong, it will not reduplicate completely (due to high-ranking *LHDIP). That leaves the two likely candidates seen here. Candidate (a) reduplicates the first vowel, which is a mid vowel, and thus loses to the second candidate, which only has a low vowel peak. This is a violation of CONTIGUITY, but the Peak hierarchy is more important in the reduplicative domain of Nakanai than contiguity.

In the tableau below are some of the vowel-markedness constraints that are ranked below MAX-BR, which means that more faithful candidates will be preferred, in general, to those candidates that better satisfy the markedness constraints.

(17) MAX-BR >> CONTIG, *DIPH

/RED+pati/	MAX-BR	CONTIG	*DIPH
a. pi.pa.ti	**!	**	
b. pa.pa.ti	**!		
☞c. pai.pa.ti	*	*	*

In this tableau the winning third candidate violates CONTIGUITY, by skipping the consonant, and *DIPH. But candidates (a) and (b) lose on their further MAX-BR violations, so it does not matter.

The rankings shown here are as follows (we will see later that MAX-BR dominates *P/i,u):

(18)
IO-Faith
egi
*LHDIP ONSET *P/i,u
yt g
MAX-BR *P/e,o
ty eg
*DIPH CONTIG *P/a

3.4.2. Analysis in Detail: Vowel Patterns

At the top of the first ranking, as we see above, is IO-Faith (and other constraints that are never violated in Nakanai or its reduplication patterns). Then several vowel and syllabic markedness constraints dominate Max-BR; these include *LHDip and Onset, as above, along with NLV. Max-BR in turn dominates the Peak constraints, along with Contig and *Diph. So the reduplicant is maximized as much the sonority-based constraints will allow. This section will only discuss the vowel patterns; what happens to the consonants in reduplication will be covered in section 3.4.3. We will discuss in turn what happens to roots of each shape.

We will turn first to some of the simplest cases. CVV roots reduplicate by patterns 2-4, and this ranking will be able to choose whether they take pattern 3 or pattern 4 (pattern 2 will be discussed in section 3.4.4.). Pattern 4 is CVV reduplication, and is thus equivalent to complete reduplication of these roots by definition (so there is no need to explain why these roots don't take pattern 1 reduplication). Tableau (19) shows a CVV root which takes pattern 3 (CV_2) reduplication:

(19) CVV root: Pattern 3 (CV_2)

/RED+sio/	IO-Faith	*LHDip	Max-BR	*P/i,u	*P/e,o	Contig
a. sio.sio		**!		*	*	
b. si.sio		*	*	*!		
☞c. so.sio		*	*		*	*
d. so.so	*!				*	

This CVV root is one which actually contains a diphthong of rising sonority in the base. Thus complete reduplication, as in the first candidate, is out because it would create an additional violation of *LHDip. Candidate (d), in which the dispreferred diphthong is left out of both the reduplicant and the base, is shown to be ruled out by the high-ranking IO-Faith. This leaves candidates (b) and (c), which each violate Max-BR once by leaving out one of the vowels in the reduplicant. The third candidate wins because of the Peak hierarchy, which prefers a mid vowel nucleus to a high nucleus, even though the candidate violates Contig and *Diph.

A CVV root which will take complete pattern 4 reduplication has to have the opposite vowel pattern, as seen in (20), since the Peak constraints would otherwise be crucial.

(20) CVV root: Pattern 4 (CVV)

/RED+bau/	*LHDip	Max-BR	*P/i,u
☞a. bau.bau			*
b. bu.bau		*!	*
c. ba.bau		*!	

In such a root, the diphthongs in the base and reduplicant do not violate the high-ranking diphthong markedness constraint. Thus the completely faithful reduplication of candidate (a) is preferable to the ones which leave out either vowel. It is clear from the two tableaux here that the set of CVV roots which take pattern 3 reduplication is entirely distinct from the set of roots which take pattern 4 reduplication, based on the vowel pairs.

This same distinctness of root types is seen in the V-initial roots. We will start with the partial reduplication, pattern 5:

(21) V-initial root: Pattern 5 (VC)

/RED+abi/	*LHDIP	NLV	ONS	MAX-BR	*P/i,u
a. a.bia.bi	*!		*		*
b. a.bi.abi			**!		*
☞c. a.ba.bi			*	*	
d. ai.a.bi			**!	*	*
e. aa.bi		*!	*	**	

The first candidate loses because of the dispreferred diphthong created by reduplication. One simple solution to the diphthong problem is used by candidate (b), which resyllabifies the complete reduplicant; this adds an ONSET violation, though, and loses. The successful candidate (c) only omits the second vowel from the reduplicant. Candidate (d) avoids a rising-sonority diphthong by dropping the consonant and keeping the high vowel in an initial, licit diphthong; this loses on its additional ONSET violation. Eliminating more of the reduplicant leads to a prohibited long vowel and ONSET violations in candidate (e). Another possible candidate, which has no ONSET violations, would be *ba.bi*; this fatally violates the L-ANCHOR constraint, by leaving out the first segment of the base. Thus even when better syllable shape would otherwise be achieved, reduplication always copies the first segment of the base.

To contrast with (21), here is a V-initial root with the higher vowel first, which will take complete pattern 1 reduplication:

(22) V-initial root: Pattern 1 (VCV)

/RED+osa/	L-ANCH	*LHDIP	ONS	MAX-BR	*P/e,o
☞a. o.sao.sa			*		*
b. o.so.sa			*	*!	*
c. sao.sa	*!			*	
d. oa.o.sa		*!	**	*	*

With a root of this shape, there is nothing wrong with the complete reduplication in candidate (a) besides the common ONSET violation and a mid vowel peak in the reduplicant. The other candidates all violate something higher, such as MAX-BR, L-ANCHOR, or *LHDIP. It is clear that the vowel pattern of the root determines which of these two reduplicative patterns a V-initial root will take.

Tableaux (21) and (22) show that V-initial roots which have the higher vowel first reduplicate completely (*osaosa*), and ones with the lower first reduplicate partially (*ababi*). This pattern should then be compared to the result of reduplicating CVV roots, as in (19) and (20): there the roots with the higher vowel first reduplicate partially (*sosio*), and the ones with the lower vowel first reduplicate completely (*baubau*). The reason why the vowel patterns have the opposite effect in V-initial roots from those in consonant-initial roots is demonstrated in these tableaux: in V-initial roots an offending rising-sonority diphthong would be created between the final reduplicated vowel and the first base vowel, i.e. at the reduplicant-base juncture (*_ab**ia**bi_), whereas with consonant-initial roots diphthongs remain or are created between the first and second vowels of the reduplicants (*_**sio**sio_ and *_**pia**pita_). Separate syllabification of the vowels is not a viable option, due to ONSET, so rising-sonority diphthongs are avoided by the non-reduplication of the higher vowel. The diphthongs are not avoided by deletion of a vowel in the base (*_ab**i**bi_), because IO-Faith is ranked above MAX-BR and because the higher vowel is dispreferred by the Peak constraints. So roots which can’t take diphthongal reduplication end up reduplicating the lower vowel, because of the Peak constraints.

We now turn to obstruent roots (roots with two obstruent consonants), which take reduplication patterns 2-4. Using this ranking, we can predict whether they will take pattern 3 (CV_2) or pattern 4 (CV_1V_2) instead. That is because roots that reduplicate by pattern 4 must have vowels that form an allowable diphthong, that is (V_1V_2) must not be a rising-sonority diphthong; roots whose vowels would form such a diphthong reduplicate by pattern 3. This result is consistent with what we saw above in (19)-(22), and shows that this ranking is able to sort obstruent roots into two classes based on their phonological shape, too. (See section 3.4.4 for pattern 2 reduplication). Tableau (23) shows an obstruent root which takes pattern 3.

(23) Obstruent root: Pattern 3 (CV_2)

/RED+pita/	*LHDIP	MAX-BR	*P/i,u	CONTIG
a. pia.pi.ta	*!	*	*	*
b. pi.pi.ta		**	*!	
☞c. pa.pi.ta		**		**

As before, complete reduplication of this obstruent root is not a viable option because of consonantal restrictions to be discussed later. Leaving out only the second consonant produces an illicit diphthong in candidate (a). We are left with two candidates that violate MAX-BR twice. So the vowel-picking Peak constraints are active, favoring the one which reduplicates a lower vowel; the winning candidate (c) violates CONTIG twice, as it skips two segments between the required leftmost segment and the favored low vowel, but that is more acceptable than the higher vowel. Thus pattern 3 obstruent roots are those in which the lower vowel is final.

In pattern 4 obstruent roots, the vowel pattern is opposite: the lower vowel is first.

(25) Obstruent root: Pattern 4 (CVV)

/RED+pati/	*LHDIP	MAX-BR	*P/i,u	CONTIG
☞a. pai.pa.ti		*	*	*
b. pa.pa.ti		**!		
c. pi.pa.ti		**!	*	**

The first candidate, in which the second obstruent is omitted from the reduplicant, creates a legal diphthong. Thus it wins over candidates (b) and (c), which violate MAX-BR twice. The ranking of MAX-BR and the vowel-selecting constraints means that roots with the lower vowel first or those with vowels of equal height will reduplicate with the first consonant plus a diphthong (pattern 4), while those with the lower vowel second will reduplicate with the first consonant and the second vowel (pattern 3).

Having now discussed CVV roots, V-initial roots, and obstruent roots, we have the sonorant roots left to cover. Sonorant roots, which take patterns 1-5, are slightly more complicated than the other root types. Below I show that a sonorant root can take pattern 1 (complete) reduplication, as in (25).

(25) Sonorant root: Pattern 1 (CVCV)

/RED+ligi/	NLV	MAX-BR	*P/i,u	CONTIG
☞a. li.gi.li.gi			**	
b. lii.li.gi	*!	*	**	*
c. li$_2$.li.gi		*!*	*	

The fully reduplicated root in candidate (a) violates almost nothing except the high vowel Peak constraint. This is just the example that shows MAX-BR must be ranked above all of the Peak vowel constraints, so that leaving anything out of the reduplicant is worse than having a high vowel nucleus. The other two candidates fail on MAX-BR or the constraint against long vowels. The problem now is that there is no reason to reduce a sonorant root at all in reduplication, so far. No consonantal constraint forces the loss of a consonant, so the vowels are never in contact to allow any other markedness constraints to become active. We will discuss in the next section how the sonorant roots ever end up with less than complete reduplication.

Now I would like to summarize what has been established so far. Nakanai reduplication is very sensitive to two main types of sonority constraints: one that regulates diphthongs (i.e. good sonority slopes over nuclei); and one that selects the best syllable peaks (i.e. high-sonority nuclei). Also playing a role are a few general markedness constraints, such as the prohibition against long vowels, and a preference for syllables with onsets. These constraints interact interestingly with CONTIGUITY, since segments are skipped in reduplication to satisfy them; Alignment and Anchoring are highly-ranked, however, unlike in Nuxalk. So we have seen that it is possible to predict which of several different reduplicative patterns these roots will take, using the Peak and diphthong constraints:

(26)CVV roots:	CV_1V_2 reduplication when V_1 more sonorous than V_2	(*baubau*)
	CV_2 reduplication when V_2 more sonorous than V_1	(*sosio*)
V-initial roots:	V_1CV_2 reduplication when V_2 more sonorous than V_1	(*osaosa*)
	V_1C reduplication when V_1 more sonorous than V_2	(*ababi*)
Obstruent roots:	CV_1V_2 reduplication when V_1 more sonorous than V_2	(*paipati*)
	CV_2 reduplication when V_2 more sonorous than V_1	(*papita*)

3.4.3. Analysis in Detail: Consonant Patterns

In this section we will explain why and when sonorant roots and obstruent roots reduplicate partially. I'll begin with the sonorant roots. As seen above, the constraint set we have been using works fine to produce complete reduplication of a sonorant base. But as yet there is no reason to reduplicate less of it. This is where the Prosodic Target constraint defined in section 2 comes in (Gafos 1997). For all of the non-sonorant roots seen in the last section, the Prosodic Target, X, can be nil, as the markedness constraints are clearly able to determine the size of the reduplicant. Thus ProsTarg(0) is part of the constraint ranking above, and is vacuously satisfied by whatever size reduplicant appears. This same target can also be seen in the complete reduplication of the sonorant root seen above.

The sonorant roots which reduplicate partially do not differ in any significant way from those that reduplicate completely, so it must be lexically marked which ones reduplicate which way. I suggest that that should be done by marking some of them as taking an affix with ProsTarg(0), and others as taking an affix with ProsTarg(Syllable). Once that parametric variation is done, the rest of the constraint ranking is exactly the same. To force some sonorant roots to reduplicate partially, it is only necessary for the ProsTarg constraint to dominate Max-BR; this makes reduplicating more than a syllable a worse violation than failing to reduplicate part of the base. A sonorant root that takes partial reduplication is shown below:

(27) Sonorant root: Pattern 3 (CV_2)

/RED+sile/	*LHDip	PT(S)	Max-BR	*P/i,u	*P/e,o
a. si.le.si.le		*!		*	*
b. sie.si.le	*!		*	*	*
c. si.si.le			**	*!	
☞d. se.si.le			**		*

Just as above, the vowel patterns in sonorant roots determine the exact shape of the reduplicant, once the general size is determined. So this root, with the highest vowel first, takes CV_2 reduplication. Candidate (a), which reduplicates the base completely, violates the ProsTarg constraint; constraint (b), which reduplicates both vowels, creates an illicit diphthong; and candidate (c), which reduplicates only the first vowel, is out because that is the higher vowel. Thus candidate (d), which reduplicates the lower second vowel, wins. Unsurprisingly, a sonorant root with the opposite vowel pattern takes CVV reduplication, as below:

(28) Sonorant root: Pattern 4 (CVV)

/RED+tari/	*LHDip	PT(S)	Max-BR	*P/i,u	Contig
a. ta.ri.ta.ri.		*!		*	
☞b. tai.ta.ri			*	*	*
c. ta.ta.ri			**!		

In this tableau, the most faithful candidate which reduplicates one syllable is (b), the winner. Candidate (a) violates the size constraint, and candidate (c) fatally violates Max by reduplicating less of the root. Thus it is clear that this constraint set predicts whether a sonorant root will take reduplicative pattern 3 or 4, based on the vowels in the root, just as well as it predicted the choices for other roots. The only difference is that sonorant roots with a ProsTarg(0) reduplicant can reduplicate completely (CVCV), and so roots that take patterns 3 or 4 need to take an affix with a syllable as the prosodic target.

Now, the other question left unanswered in section 3.4.2. was why obstruent roots only take the partial reduplicative patterns (and not pattern 1, complete CVCV reduplication). This fact is clearly illustrated by the following table, which is based on the data in Johnston (1980). The numbers show how many bases containing the relevant pair of consonants take pattern 1 vs. any other pattern, since patterns 2-5 involve only the first of the consonants being reduplicated. For example, the cell for roots with the consonant pair *p-t* shows that no bases of this shape take pattern 1 reduplication, and two take other patterns. Obstruent-obstruent cells are shaded. (*H*-initial stems are left out of this table since they only take pattern 5, e.g. *hararo*.)

(29) Reduplicative patterns by consonant pairs

C1/C2	p	b	t	d	k	g	β	s	h	m	n	l	r
p			0,2			0,1			0,1		1,0	2,1	1,0
b		0,2	0,2					0,2				4,1	3,0
t					0,1	0,3	(1),2		0,2	0,1		0,6	0,2
d												0,1	
k	0,1		0,1	0,1			0,2	0,5				3,1	3,3
g	0,1		0,1				0,2	0,1				2,0	0,4
β						0,1						2,1	2,0
s	0,1	0,1			0,1		0,1	0,2			1,1	0,3	
m	0,1		0,1		0,1	1,2		0,1		0,1		3,0	1,0
n					0,1								
l			1,1		2,2	8,0	1,0		0,1	2,0		0,2	
r			1,0			1,0	3,0	1,0					

Obstruent-obstruent pairs (shaded) add up to 1 root reduplicating by pattern 1, 40 by other patterns; obstruent-sonorant pairs add up to 23 roots taking pattern 1, 22 taking other patterns; sonorant-obstruent pairs add up as 19 roots taking pattern 1, 10 taking other patterns; and sonorant-sonorant pairs add up with 6 roots taking pattern 1, 3 taking other patterns. Obstruent roots simply don't reduplicate completely, whereas roots with at least one sonorant (particularly the very common *l*) often do reduplicate with both consonants. The only obstruent pair that shows complete reduplication is *t*-β, and that example is optional; the root in question takes either pattern 1 or pattern 2 reduplication.

There does not seem to be an active constraint against obstruents generally in the language; Nakanai has a fair variety of obstruents and uses them, though perhaps the sonorants *r* and *l* are more frequent. So one could write a constraint against obstruents in reduplicants, or a constraint against obstruents that only emerges in the reduplicative domain (the emergence of the unmarked schema). This constraint is not clearly motivated typologically. Nevertheless, these facts exist, and deserve some explanation. I favor an OCP account, using the following constraint:

(30) OCP[-son]: Adjacent [-son] specifications are prohibited

This constraint penalizes any two obstruent consonants not separated by a sonorant consonant. Through ranking, this constraint can be restricted to having an effect only in reduplication. As long as the OCP is ranked beneath IO-Faith, it will not be able to affect the inventory of roots in the language; and as long as it is ranked above Max-BR, it can compel non-reduplication of one obstruent in a reduplicant. The following tableau shows how this constraint works:

(31) Obstruent root: OCP effect

/RED+pati/	IO-Faith	L-Anch	OCP	Max-BR	*P/i,u
a. pa.ti.pa.ti			***!		*
☞b. pai.pa.ti			**	*	*
c. ai.pa.ti		*!	*	**	*
d. pa.pa.ti			**	**!*	
e. pai.pai	*!		*		*

The OCP rules out candidate (a), which reduplicates both obstruents of the base. Candidate (c) fails to reduplicate both obstruents of the base, and violates L-Anchor. Candidate (e) has the OCP overapplying to the base, which violates IO-Faith. The choice is then between candidates (b) and (d), which both leave out the second obstruent, and the normal constraint ranking favors the diphthong in (b), since it is an allowable diphthong.

The OCP[-son] appears to work fine for the obstruent roots for which it was designed. This means that this constraint-ranking can predict not only whether obstruent roots will take pattern 3 (CV_2) or pattern 4 (CVV) reduplication, but can also predict that obstruent roots will not take pattern 1 (CVCV) reduplication. Having added this constraint to our ranking at this stage in the paper, however, one might wonder whether any of the results of section 3.4.2. have been undone.

The answer to that question is no. The CVV roots, such as *bau*, will not be affected by the OCP, since the only consonant in these roots is the first one, which must surface in the reduplicant due to the L-ANCHOR constraint. Sonorant roots like *tari*, even when reduplicated as *taitari*, will also have the obstruent protected by the L-ANCHOR constraint; sonorant roots like *ligi*, reduplicated as *ligiligi*, will keep the obstruent since there is a sonorant consonant between the instances of obstruents. The only roots there might be a question about are the V-initial roots like *abi*, reduplicated as *ababi*, in which the obstruent is not protected by L-ANCHOR or a sonorant consonant. These roots can be protected, simply enough, by ranking the OCP below the ONSET constraint. Any candidate which leaves out the obstruent, like **ai.abi*, has to end up incurring more ONSET violations (or markedness violations: **aa.bi* (NOLONGVOWEL)). So we have shown that an OCP[-son] constraint can be added to the constraint ranking that has been used so far, with no ill effects, and that it will capture the generalization that obstruent roots never reduplicate completely.

In this section constraints to capture the reduplicative patterns taken by roots with two consonants have been motivated. Sonorant roots usually take the PROSTARG(0) reduplicant that the rest of the roots described take, so there is no predetermined reduplicant size. But some sonorant roots need to be specified as taking a reduplicant that is only as big as a syllable, so the parametrized PROSTARG(Syllable) is needed for them. With this constraint parametrization available, the patterns we have described so far can be produced for sonorant roots. Obstruent roots, on the other hand, can always take the PROSTARG(0) reduplicant, but some constraint needs to rule out reduplication of both obstruents in a root. This section has shown that an OCP[-son] constraint will correctly restrict reduplicants. The sonority restrictions seen in the rest of this analysis are still visibly playing a role, determining which vowels will be reduplicated in these types of roots. Also, perhaps the OCP[-son] constraint could be linked to a general preference for high sonority in the reduplicant in Nakanai. It is at least suggestive that this consonant restriction involves sonority, rather than some other property of consonants.

3.4.4. Second Ranking: Pattern 2 Reduplication

Finally, there is one reduplicative pattern in Nakanai that can’t be generated using the exact constraint ranking seen so far: pattern 2 reduplication (CV_1). This means that some type of constraint reranking is necessary for those roots that take pattern 2 reduplication.The second constraint ranking, however, only involves ranking the MAX-BR constraint a bit lower with respect to the same markedness constraints as above. Now, since different reduplicative affixes are predicted to have different Correspondence constraints regulating their relationship to the base (McCarthy and Prince 1995, Urbanczyk 1996), we can see pattern 2 as involving a different reduplicative affix. So we split up MAX-BR2, the constraint prohibiting deletion between the base and reduplicative affix 2, from MAX-BR1. And MAX-BR2 happens to be ranked somewhat lower in the same constraint hierarchy than MAX-BR1 is. Specifically, MAX-BR2 is ranked below the Peak constraints for mid and high vowels, along with CONTIGUITY and the *DIPH constraint. This means that it is very important in this pattern to select a good Peak for the reduplicant. Roots are lexically marked, then, as to whether they take reduplicative affix 1 or 2, and thus which MAX constraint applies.

We should note that V-initial roots don’t take this reduplicative pattern, simply due to their shape. The two patterns they take are covered by the first constraint ranking. So, turning first to the obstruent roots, we can see how the Peak constraints and CONTIGUITY favor reduplicating the first vowel when consonantal restrictions rule out complete reduplication (for two different root shapes):

(32) Obstruent roots: Pattern 2 (CV_1)

/RED+kapu/	OCP	*P/i,u	*P/e,o	CONTIG	*DIPH	MAX-BR2
a. kau.ka.pu		*!		*	*	*
☞b. ka.ka.pu						**
c. ku.ka.pu		*!		**		**
d. ka.pu.ka.pu	*!	*				

/RED+guvi/	OCP	*P/i,u	CONTIG	*DIPH	MAX-BR2
a. gui.gu.vi		**!	*	*	*
☞b. gu.gu.vi		*			**
c. gi.gu.vi		*	**!		**

The way I have been marking Peak violations, both vowels in a diphthong count as part of a peak, so a diphthong with a high vowel in it gets a *P/i,u mark. An alternative use of the Peak constraints would be to mark violations only for the first vowel of a diphthong, which is arguably the main vowel. The first tableau above is decided by Peak violations of the sort just described, but one should note that the same losing candidates also violate CONTIGUITY; so even if this way of marking Peak violations were not accepted, this ranking generates the right results. The first vowel is reduplicated, in both tableaux, since it is equal to or lower than the second vowel, and reduplicating more than this vowel leads to more Peak violations (or more CONTIGUITY violations).

With CVV roots, the same result is achieved:

(33) CVV roots: Pattern 2 (CV_1)

/RED+vei/	*P/i,u	*P/e,o	CONTIG	*DIPH	MAX-BR2
a. vei.vei	*!	*		**	
b. vi.vei	*!		*	*	*
☞c. ve.vei		*		*	*

/RED+giu/	*P/i,u	CONTIG	*DIPH	MAX-BR2
a. giu.giu	**!		**	
b. gu.giu	*	*!	*	*
☞c. gi.giu	*		*	*

In both of these tableaux, the Peak (or CONTIGUITY and *DIPH) violations rule out the (a) and (b) candidates in favor of the (c) candidates, which reduplicate the first vowel.

Finally, with sonorant roots we see the PROSTARG(Syllable) constraint being active to rule out complete reduplication; this constraint needs only to be ranked somewhere above MAX-BR2. First, the tableau for a root with identical vowels, in which the markedness constraint against long vowels is active:

(34) Sonorant root: Pattern 2

/RED+tolo/	NLV	*P/e,o	CONTIG	MAX-BR
a. too.to.lo	*!	**	*	*
b. to$_4$.to.lo		*	*!*	**
☞c. to$_2$.to.lo		*		**

In this table the most faithful candidate (a) violates the high-ranking constraint against long vowels; candidate (b), which retains only the final vowel, loses on its two CONTIG violations to the one which retains the first vowel. And the final sonorant root table showing pattern 2 reduplication, showing a root with a lower first vowel:

(35) Sonorant root: Pattern 2

/RED+kali/	*P/i,u	CONTIG	*DIPH	MAX-BR
a. kai.ka.li	*!	*	*	*
☞b. ka.ka.li				**
c. ki.ka.li	*!	**		**

The candidate that violates the PROSTARG constraint is not shown here; candidates (b) and (c) violate the Peak constraints fatally, as well as CONTIGUITY. The successful candidate (b) only violates MAX-BR.

As with the first ranking, the sonorant roots work almost the same as the rest, with the exception of the PROSTARG(Syllable) constraint. And the obstruent roots are uniquely affected by the OCP. But what we see in this ranking is that all of the same sonority-based constraints are active, they are just a bit more important than in the first ranking. So the pattern 2 reduplicative affix would rather have the best possible nucleus, and no marked diphthongs or CONTIGUITY violations, than reduplicate more segments of the base.

3.5. Further Issues in Nakanai

3.5.1. Output-Output Relations

There are several places in this analysis where relations between output forms seem important. One is to capture the fact that adding other affixes to a root does not change the reduplication. Roots that reduplicate can take other affixes, such as *-ti*, the perfective suffix. And the metrical foot of a suffixed word, in this language with predictable stress, is different from that of the unsuffixed form: (*tuga*) vs. *tu*(*gati*). But reduplication is aligned to the same base (or foot), *tuga*, in both of these words: *tatuga* and *tatugati*, not **tugagati*. The reduplicant in a word like *tatugati* is not correctly aligned to the foot in the reduplicated form, but it is aligned to the material that comprises the foot in the corresponding unsuffixed form of the word. Thus an Output-Output Alignment relation (to be discussed in more detail below) would capture the fact that all forms with the same root take the same reduplicative base, since the Alignment constraint could refer to the foot in the output, unsuffixed word.[9]

Another context in which an Output-Output Alignment relation is useful is to capture reduplicative Alignment in V-initial stems. Now, in the initial formulation of the Alignment constraint above, it simply claimed that the right edge of the reduplicant is aligned to the left edge of the foot. This is normally the case, with consonant-initial roots: *pa(pita).* But with V-initial roots, one can quickly see that the reduplicant is not aligned with the edge of the foot: *a.(ba.bi)*, and *o.(sao.sa)*; the reduplicant is overlapping the foot in the output reduplicated form. However, the V-initial stems all reduplicate like this, and it is clearly a function of their syllabic structure, rather than a truly different placement of the reduplicative affix. So, one might note that in the V-initial stems, the reduplicant is actually prefixed to the material that comprises the foot in the unreduplicated form: *(a.bi)* and *a.b(a.bi)*. To capture this fact, I suggest that the correct Alignment constraint aligns the right edge of the reduplicant to the segment that corresponds to the left edge of the foot in the unreduplicated form.

(36) OO-Align (RED, R, OutputFoot, L): Align the right edge of the reduplicant to the left edge of the foot of the root

This constraint postulates a relation between the reduplicated and unreduplicated output forms of a root, which allows reduplicative Alignment to be checked against the unreduplicated output. This lets V-initial roots be as well-aligned as C-initial roots.[10]

The following tableau shows how this OO-Alignment constraint would work.:

(37) V-initial root with O-O Alignment

/RED+abi/	*LHDip	NLV	Ons	OOAlign	Max-BR	*P/i,u
a. a.bi.a.bi			**!			*
☞b. a.ba.bi			*		*	
c. ai.a.bi			**!		*	
d. aa.bi		*!	*		**!	
e. a.bi.bi			*	*!		

Most of the candidates shown do not violate this new Alignment constraint. Candidate (a)

[9] There are some derivational affixes which affect reduplication, so that the reduplicated form seems to be based on the new derived root. One such is the nominalizing infix *-il-*, which creates the noun *pileho* 'death' from *peho* 'to die'. This then can reduplicate to form a concrete noun as *pileleho* 'corpse', which includes the *l* of the affix.

[10] For other work in Optimality Theory on relations between output words, see for example Benua 1995, 1997.

exemplifies the candidates which reduplicate the whole V-initial root; these will fail either on the diphthong constraint or on ONSET violations. Candidates (b)-(d) show other, less complete reduplicants. The final candidate here is one that violates the new Alignment constraint, since it reduplicates a smaller portion of the root than usual, and does not align the reduplicant to the original foot. As above, candidate (b) is the winner, since it only fails to reduplicate the high vowel.

There is one other root type for which this constraint is interesting: the *h*-stems. These are the exceptional C-initial roots which reduplicate as if they were V-initial, as in *hararo*. These roots have the reduplicants aligned differently from either the C-initial or V-initial roots, since the portion of the root they are prefixed to is not a foot in either output form: *(ha.ro)* and *ha.(ra.ro)*. So these roots violate OO-Alignment, just as they would violate the original Alignment constraint from section 3.3. In fact, to allow *h*-stems to reduplicate this way, OO-Alignment must be crucially ranked below several other constraints, whereas the regular Alignment constraint in Nakanai seems to be among the undominated constraints. Thus I suggest that these stems involve a re-ranking of the constraints; since this pattern is obviously a rare and exceptional one, this does not seem like a problem.

There is one new markedness constraint that must also be added to the ranking to produce *h*-stem reduplication:

(38) *VhV: Intervocalic *h*'s are prohibited

Since most of the few roots that take this type of reduplication begin with *h*, and *h* is a disappearing consonant in Nakanai, this constraint seems like a plausible way to produce this pattern.[11] This is one of the constraints that must dominate the Alignment constraint, as shown by the following tableau:

(39) *H*-stem reduplication

/RED+haro/	NOCODA	*VhV	*LHDIP	ONS	OOALIGN	MAX-BR
a. ha.ro.ha.ro		*!				
b. har.ha.ro	*!					*
c. ha.roa.ro			*!		*	
d. ha.ro.a.ro				*!	*	
☞e. ha.ra.ro					*	*

Candidate (a), which reduplicates the base completely, violates the markedness constraint against intervocalic *h*. The second candidate reduplicates most of the base, but falls foul of the NOCODA constraint, which is unviolated and thus undominated in the language as a whole (except in some borrowed words). Candidate (c) initiates the strategy of taking a base smaller than the whole foot and misaligning it after the initial *h*; however, while eliminating the intervocalic *h* problem, this strategy creates an unpreferred diphthong between the reduplicant and base. Candidate (d) takes

[11] For a more dramatic example of a constraint against intervocalic *h* interacting with reduplication, see the analysis of Javanese in McCarthy and Prince 1995.

the same strategy, but syllabifies the output differently. This avoids the diphthong, but leads to a fatal ONSET violation. Thus the winning final candidate takes pattern 5 reduplication, failing to reduplicate the final vowel of the vowel-initial base and putting the initial *h* out of the way by misalignment. All *h*-stems reduplicate this way, though one of them has a vowel pattern that does not allow candidate (c) to be ruled out; this could be another output-output identity relation, in that all of the *h*-stems want to reduplicate alike, regardless of vowel patterns.

These tableaux show that using the new Output-Output Alignment constraint ensures the selection of the correct winning candidate for V-initial roots, and can be reranked to produce the odd pattern of *h*-stems. Other roots will not reduplicate in this way, as in **pitaita*, without the coercion of a high-ranking constraint like *VhV.

3.5.3. Predictability

The last issue to mention is that there is some predictability in Nakanai reduplication which is not captured by this analysis. This analysis predicts whether roots will take patterns 1, 3, 4, or 5 through one constraint-ranking, and generates pattern 2 with the other (with a different reduplicative affix). The assignment to constraint-rankings is not predicted, and should be lexically determined and thus random. But there are some generalizations about which roots will fit into which constraint-ranking. The overwhelming majority of the roots taking pattern 2 reduplication are those with the lowest vowel first, e.g. *kakapu*. So a consonant-initial root which has a higher vowel first will be predicted to be in the class that takes the first major constraint ranking (e.g. *papita*, not **pipita*). The vowel-initial roots never take pattern 2 reduplication due to their shape, so they are in the class which takes the first constraint ranking. Roots with identical vowels are assumed to be taking pattern 2 (CV_1) reduplication (rather than pattern 3, CV_2), so these roots will be placed into the second constraint-ranking set. The roots which do not fit any of these descriptions are the ones for which we are unable to predict which constraint ranking they will take. Thus roots with a lower first vowel or vowels of equal height fit into both lexical classes.

3.6. Conclusion

Nakanai has a complex set of reduplicative patterns, with phonological conditions partially determining which pattern roots will take. These phonological conditions are expressed as plausible universal sonority constraints. In particular, vowel-markedness constraints seem to be very important in reduplication; these include constraints against rising-sonority diphthongs, which produce a dispreferred sonority slope through the syllable, along with the Peak-markedness constraints that favor higher-sonority vowels as nuclei. There is also an OCP constraint on obstruent consonants, which could be linked to a sonority preference. Using these constraints, along with the Correspondence Theory faithfulness constraints, the reduplication patterns are partially predictable. These sonority constraints are general constraints used in the syllabification systems of most languages, so their use in reduplication is not an unexpected or stipulative result, but an extension of a very general system to a particular domain.

The fact that there are near-minimal pairs of roots that nevertheless take different reduplicative patterns means that there is some unpredictability in the system. This is captured in part by the parametric PROSTARG constraint, which allows roots of one type to vary in what the preferred reduplicant size is. Also, there are slightly different constraint rankings for two reduplicative affixes: one creates pattern 2 reduplication, and the other creates patterns 1, 3-5.. This is because these affixes vary in how highly their MAX constraints are ranked with respect to the markedness constraints.

Finally, the effects of the sonority constraints on Nakanai reduplication are interestingly different from those in Nuxalk, which will be described in more detail in the following section. In particular, the Alignment and Anchoring constraints are highly ranked in Nakanai, while the CONTIGUITY constraint is low-ranked.

4. Nuxalk Overview

Nuxalk (Bella Coola) is a Salish language spoken in British Columbia (Bagemihl 1991). One feature that has been of particular interest to linguists is its odd or nonexistent syllable structure (Hoard 1978, Newman 1947, 1971, Nater 1984). It is a language with many consonants, and words can contain few or no vowels: *kmɬ* ‘fin’, *t’χt* ‘stone’. Also, though a few stress generalizations exist, for many words there is no constant stress pattern. Thus some researchers have argued that the language lacks syllables, others that it allows almost any segment to be the nucleus of a syllable. Bagemihl’s excellent article on the subject uses facts from reduplication and a few other domains to argue that the language actually has very simple syllables, with a maximum CCVVC shape.

I agree that the reduplicative facts suggest a certain syllable shape, and will show that some interesting sonority generalizations account for these facts. And this paper is restricted to a discussion of reduplication, rather than a complete investigation of syllabification in the language. I would like to point out, however, that reduplication often copies an unmarked subset of what a language may allow in the way of prosodic and featural structures. This is what the emergence of the unmarked ranking schema is all about: markedness constraints becoming active in the reduplicative domain, though they may not be seen in the language as a whole. Thus Bagemihl’s argument that the only possible syllable structure of Nuxalk is that which is seen in reduplication is not a completely convincing one.

This analysis of Nuxalk reduplication will focus on the facts that relate to sonority, which I believe are the major ones. Reduplication in Nuxalk is usually prefixal, as in *silsilin* ‘kidney’, but sometimes can be an infix, as in *t’ksnsn* ‘shoot with bow’ and *p’ɬaɬa* ‘wink’. The generalization here is that the cases where reduplication is infixed are cases where the first sonorant consonant or vowel of the root is several segments inside the word. Thus I believe the Peak constraints are active here. The reduplicative affix can be moved into the word in search of a nucleus that is sonorous enough: a sonorant consonant or vowel. So the Alignment constraint which makes reduplication a prefix is ranked between *P/obstruent and *P/sonorant, so that reduplicating an obstruent peak is worse than mis-Aligning the prefix.

4.1. Introduction to Nuxalk

The segmental inventory of Nuxalk is well-stocked with consonants, as in other languages of the region:

(40) Phonemes

	Labial	Alveolar	Lateral	Velar	Uvular	Glottal
stop [+glot]	p p’	t, c t’, c’		k, k^w k’, k'^w	q, q^w q’, q'^w	ʔ
fricative [+glot]		s	ɬ ɬ’	x, x^w	χ, $χ^w$	
sonorant	m	n	l			
vowel		i/y	a	u/w		

The high vowels and sonorant consonants alternate predictably between syllabic and non-syllabic uses. As shown, many consonants have secondary articulations such as rounding and glottalization. Descriptions of the segments vary somewhat, but Newman (1947, 1971) and Nater (1984) both describe the “velar” segments as being pre-velar or palatovelar, while the “uvular” segments are distinctly postvelar; so these phonemes may not be placed exactly as the phonetic symbols used would suggest.

In the introduction above I mentioned the controversy over syllables in Nuxalk. This paper does not solve it, but shows that it is not necessary to assume simple syllabification in the language in order to make the reduplicative facts work out. The reduplicative domain is one in which unmarked phonological elements should emerge, so it is not unlikely that fairly canonical syllables would emerge in reduplication. On the subject of syllabification it may be interesting to compare Nuxalk with Lushootseed, another Salish language; Urbanczyk (1996) makes a strong case for Lushootseed syllables having voiceless vowels, when only obstruents are present. I have seen no similar evidence in Nuxalk, but suggest that a comparison of the two languages might be useful for settling the Nuxalk syllabification questions.

4.1.1. Reduplication in Nuxalk

Nuxalk reduplication is used to express diminutive and continuative meanings, as well as marking some derived forms and compounds (the addition of some affixes, as well as the formation of compound nouns, can trigger reduplication). The form of the reduplicant does not seem to vary with the meaning.

Phonologically, Nuxalk reduplication seems to be both complicated and partly lexical. That is, not all alternations are predictable. However, I think some predictable aspects have not been clearly brought out in earlier work. There are a few sub-patterns that occur in small numbers of roots, which will be discussed in section 4.5; the major patterns of reduplication are shown in (41):

(41) Reduplicative patterns

1. CV		
tup	tutup	‘trout’
smɬk	smsmɬk	‘fish’
stlk’^{w}a	stltlk’^{w}i	‘jelly fish’
p’ɬa	p’ɬaɬa	‘wink’
2. CVC		
silin	silslin	‘kidney’
tux	tuxtux	‘unwind’
mnɬk^{w}a	mnɬmnɬk^w-ni	‘hair’
k^{w}paɬ	k^{w}paɬpɬ-i	‘liver’
3. V		
p’wi	ʔup’wi ‘halibut’	
t’ixɬala	ʔit’ixɬalay	‘robin’

One thing to note about these patterns is the size of the reduplicant. Although these three patterns are different, none of them reduplicates more of the base than could be called a syllable. And whether or not these groups of segments constitute syllables within the base, I believe they constitute syllables in the reduplicant.Thus the setting of the Prosodic Target constraint which is needed for all of these patterns is ProsTarg(Syllable). As I will show, there is a certain amount of predictability between the CVC and the CV patterns, so some roots will have an unspecified syllable as their prosodic target, and markedness constraints will decide. Some roots, however, do not pattern with the others in predictability, and for those I will have to specify whether the target is a bimoraic or monomoraic syllable.

Other notable facts about these patterns include the variation in reduplicant nuclei between vowels and sonorant consonants, and the variation in the position of the reduplicant, from prefix to infix. Both of these will be explained by this analysis, along with some discussion of auxiliary processes. What will not be explained are two of the other processes that interact with reduplication: syncope and vowel-lengthening (and vowel-shortening). These processes can occur along with reduplication, to express the continuative and diminutive meanings, and without it as well.

(42) Other processes

yaɬ	yaɬyaaɬk	‘do too much’
kap’ay	kaakp’ay-i	‘humpback salmon’
sq^{w}caals	sq^{w}cacls-i	‘cheek’
niq’χ	niiq’χ-i	‘otter’

In Lushootseed (Urbanczyk 1996), syncope in bases occurs in a distinct environment: only between consonants that fall or are equal in sonority. This fact can be linked to a related process of vowel reduction and some important stress facts, and is convincingly explained. In Nuxalk, though, a similar restriction to certain environments is not found; syncope occurs just as much between consonants that fall in sonority as it does between other consonants, and there are no stress and reduction facts to provide further evidence. So these processes will be ignored in the rest of this analysis.

4.2. Main Sonority Generalization

In the patterns above, we noted that each reduplicant contained at least a vowel or sonorant consonant; this holds true through all of the reduplication in the language (obstruent-only roots either don't reduplicate or do so specially; cf. section 4.5). Also, the reduplicant was sometimes a prefix, and sometimes an infix. These facts can be linked: in every case where the reduplicant is infixed, it appears before the first sonorant or vowel in the word. In fact, it appears one segment before it, and that segment is always reduplicated as well. This suggests that the language looks for the first segment that is sonorous enough to be a nucleus, and reduplicates it with the preceding segment as an onset. The next section will discuss the choice between CV and CVC reduplication, so we will ignore that decision for now. This section focuses on what nucleus is reduplicated, and where the reduplicant ends up.

The Peak constraints will be very useful for this phenomenon. These rank all segments in the language as to their relative sonority, which correlates with their fitness to be syllable nuclei. Thus all of the specific *P/obstruent constraints are ranked above all of the *P/sonorant constraints, which are ranked above the *P/vowel constraints.

(43) *P/obstruent >> *P/sonorant >> *P/vowel

If the constraint which forces the reduplicant to be a prefix is ranked between the *P/obstruent and *P/sonorant constraints, then the language would prefer to mis-Align the reduplicant in order to avoid having an obstruent nucleus in the reduplicant. The particular Alignment and Anchoring constraints for Nuxalk reduplication are as follows:

(44) ALIGN (RED, R, Word, L): Align the right edge of each reduplicant to the left edge of the word
L-ANCHOR: the element at the left edge of the base has a correspondent at the left edge of the reduplicant

So, the reduplicant should be a prefix and Anchored to the base, much as in Nakanai. As mentioned earlier, it is crucial that L-ANCHOR be understood here with respect to the definition of the base of McCarthy and Prince (1995) (used also in Urbanczyk 1996), which is that the base is the string adjacent to the reduplicant. Thus by infixing a reduplicant it is possible to violate Alignment, but not L-ANCHOR.

PROSTARG(Syllable) is highly ranked, so that the reduplicant is always syllable-sized. This constraint then must dominate MAX-BR, which would demand complete copying of the root (or the portion of it that is the base). Also, CONTIGUITY is high-ranking, since reduplication doesn't take non-Contiguous portions of the base. In the major patterns, there is no epenthesis, so DEP-BR is undominated. Reduplication always takes an onset, except in a pattern to be discussed later, so ONSET is high-ranking. And finally, reduplication never takes more than one segment as an onset, so *COMPLEX is highly ranked. In the tableaux below, Peak constraint violations will only be assessed for reduplicants, since I will remain neutral about the syllabification in the base.

So, let us see now how these constraints work to generate the correct patterns. A root with an infixing reduplicant is shown below[12]:

(45) Infixing case

/RED+st’q^wlus/	L-ANCH, CONT	*P/obstr	ALIGN-Pre	*P/sonorant	*P/vowel
a. st’st’q^wlus		*!			
b. st’q^wt’q^wlus		*!	*		
☞ c. st’q^wlq^wlus			**	*	
d. st’q^wlulus			***!		*
e. q^wlst’q^wlus	**!			*	
f. slst’q^wlus	**!			*	

The first candidate (a) begins reduplication at the beginning of the word, as one might expect. This is a problem, however, since the word begins with a cluster of obstruents. The first and second candidates are ruled out by high *P/obstruent violations, though the second has begun infixing the reduplicant in search of a better nucleus. The successful third candidate, (c), which moves inward enough to reach the sonorant *l*, violates the Alignment constraint, but has a good enough nucleus. Moving the reduplicant farther inward, as in (d), gets to an even better nucleus, but is less close to being a prefix, so its Alignment violations rule it out. Candidate (e) keeps the sonorant as nucleus, but tries to satisfy the Alignment constraint at the expense of L-ANCHOR. Here we see what the L-ANCHOR constraint does. The reduplicant infixes to find a good enough nucleus, and also to be directly in front of what it reduplicates, so that the left edge of the reduplicant is Anchored to the left edge of the new base. Thus satisfying Alignment at the expense of Anchoring is not preferred in this language. Finally, candidate (f) tries to fix the Anchoring and Alignment problems by staying a prefix and reduplicating the first consonant of the word, but has to violate CONTIGUITY in order to keep a good nucleus. Other candidates that one could construct violate high-ranking constraints mentioned above. For example, the candidate *st’q^wlst’q^wlus* violates the ban on complex syllable onsets, *COMPLEX. And *sist’q^wlus* is ruled out by DEP, since infixation is better than epenthesis.

So what we are seeing in Nuxalk reduplication is a type of threshold effect. Unlike in Berber (Dell and Elmedlaoui 1985, 1988), not all consonants are allowed as nuclei; the obstruents are not good syllable peaks. But the sonorant consonants, while they may not be as good as the vowels, are good enough to satisfy the reduplicant’s need for a sonorous nucleus. The reduplicant infixes as far as it needs to in order to find a good nucleus, but the tension between Alignment and the Peak constraints keeps it from moving farther to find the best possible nucleus (a low vowel, for example).

Now, what about a root that doesn’t begin with a consonant cluster? The next tableau shows that this ranking also captures prefixal reduplication:

[12] Some of the roots with initial obstruent clusters begin with *s*; this is apparently a nominalizing prefix (Nater 1984). However, roots with monomorphemic clusters reduplicate the same way, so this does not seem to be a morpheme-specific pattern.

(46) Prefixing case

/RED+milix^{w}/	L-ANCH, CONT	*P/obstr	ALIGN-Pre	*P/sonorant	*P/vowel
☞ a. milmilix^{w}					*
b. lix^{w}milix^{w}	**!				*
c. milix^{w}lix^{w}			**!		*
d. mlmilix^{w}	*!			*	

For a root of this shape, the prefixing reduplication comes out quite naturally. The coda issue will be discussed later, but this tableau shows how a base with a vowel or sonorant nucleus at the beginning of the word reduplicates. Candidate (c) tries infixing the reduplicant, but since the prefix violated few markedness constraints, the infix loses on Alignment. Candidates (b) and (d) try reduplicating material from further in the base as a prefix; this violates either L-ANCHOR or CONTIGUITY, both fatal violations. So the prefixing reduplicant in (a) wins.

I believe this section has shown two things. One is that sonority generalizations, interacting with Correspondence constraints, can account for the segmental content and position of reduplicants in Nuxalk. The other is that this account works without discussing syllabification in the language in general, and can stand equally well whether the language has only simple syllables (as in Bagemihl 1991) or complex, exhaustive syllabification of consonants (as in Hoard 1978). This is because of the emergence of the unmarked ranking schema. The Peak constraints, which interact here with the Alignment of the reduplicative affix, could be ranked below the constraints which govern the syllabification of output words in Nuxalk. The phonological generalizations seen in reduplication could be ones that are not active (or rarely seen to be active) elsewhere in a particular language. The separate Correspondence constraints on the Input-Output relation and the Base-Reduplicant relation allow markedness constraints to only affect one domain.[13]

4.3. Minor Sonority Generalization

Nuxalk reduplication has a set of coda conditions which can also be captured using a partially sonority-based account. The fact is that some segments that exist in the language, which can be reduplicated as onsets in reduplicants, are never found in the codas of reduplicants. These segments are less sonorous and more marked than the segments which are allowed in codas. So I see this as an emergence of the unmarked effect, which is linked to the ideal sonority slope of a syllable. The ideal syllable has a slow fall in sonority after the nucleus or no fall at all (the NOCODA effect), so restricting codas to more sonorous segments makes sense (Clements 1990,

[13] Urbanczyk (1996) makes a good case for exhaustive syllabification of consonants in Lushootseed, with voiceless vowels showing up in particular contexts. The present account of reduplication would still work if this type of syllabification was found in Nuxalk (assuming that voiceless vowels are quite marked, and that the markedness constraint against them could dominate B-R Correspondence constraints).

Zec 1995). And codas are less privileged positions than onsets (Lombardi 1996, Beckman 1996), so the restriction to less marked segments is also motivated.

4.3.1. Facts and Generalizations

The codas that are found in reduplication, with a few exceptions, are comprised of only the following segments:

(47) Allowable codas: l,ɬ, x, s, n

milmilix^{w}-ɬp	‘bear berry plant’
q’iɬq’ɬ	‘scratch’
tuxtux	‘unwind’
t’q^{w}usq^{w}s-i	‘gnat’
winwints	‘sandpiper’

There are a few things that are noteworthy about this particular set of codas. One is that no stops or affricates are included, as well as no segments with secondary articulations (and this is a language with numerous secondarily articulated consonants). Because of these restrictions, in many cases we can predict whether a root will take CVC or CV reduplication. For example, since no stops are allowed as codas in reduplication, a base with a stop after the correspondent of the reduplicated nucleus will not end up with that as a coda: *tutup*, not **tuptup*. It is possible to predict reduplicant size this way in numerous cases. But there are some exceptions.

First of all, some roots have segments that could be reduplicated as codas, but simply aren’t: *cacaaɬi*, *smsmɬki*. These roots are fairly common. This means that we have a generalization about codas that only holds in one direction; that is, we can tell which roots won’t reduplicate with codas, but not which ones will. Still, a one-way generalization means that there is some predictability, which should be explained. I believe the best way to handle the exceptions just noted is to lexically specify that these roots take an affix with ProsTarg(Syllable$_{\mu}$); that is, they take a reduplicative affix whose preferred size is a one-mora syllable, rather than two. Secondly, some roots which do not contain possible codas (and a few that do) end up with a coda *n* in the reduplicant: *sunsupt*, *cuncut*, *cpunpuus*. These roots are quite uncommon: about 11 roots take this pattern in some form. And these can be lexically specified as taking ProsTarg(Syllable$_{\mu\mu}$), so that they demand a CVC reduplicant; perhaps the markedness constraints are ranked for these affixes so that *n* is the best epenthetic consonant.

These exceptional patterns are not terribly interesting. What is very interesting, though, is a robust generalization about the Nuxalk velar and post-velar consonants (as noted above, the velars may be more fronted than in some languages): many of them reduplicate as the velar fricative *x* in coda position.

(48) Velar and uvular coda patterns

yaxyaaki	‘mountain goat’
nixnik’	‘cut’
sixsik’ʷ	‘pull’
nixniq’χm	‘to have cramps’
sc’ixc’ixʷtaɬp	‘horsetail plant’
ɬaxɬ’aq’ʷɬp	‘Douglas fir tree’
paxpaχukʷ	‘he’s always afraid’

This is just a sample showing that velar and uvular consonants with and without secondary articulations normally reduplicate as the velar fricative, with only a few exceptions (a similar effect on a smaller scale is seen in Nisgha (Shaw 1988)). On one view (McCarthy 1991), uvular segments are segments with both velar and pharyngeal articulations. If this is so, then all of the cases shown are ones in which secondary articulations are lost while the velar place remains, and non-continuant segments are spirantized. So the same constraints that are active in picking allowable codas generally (i.e., constraints against stops and secondary articulations) affect segments with velar articulations, too; it seems that velars are affected in a special way since they all neutralize to *x*, while a coronal *t* doesn’t neutralize to *s*.

There are several ways to explain this fact about velars. One tack would be to claim that it is easier to lenite velars than other segments, in this language at least. The idea that segment types can be ranked as to the likelihood of their undergoing some process like lenition has been explored in work by Foley (as discussed in Smith 1981), for example. A way to capture that in constraints would be to parametrize a constraint against lenition, IDENT[cont], so that separate constraints would regulate identity for the feature [continuant] between corresponding velar segments, corresponding coronal segments, and so on.

In Nuxalk, the uvulars at least seem somewhat unstable already in circumstances connected with diminutive and continuative formation. For example, Newman (1947, 1971) discusses various lenition and fortition processes in roots that may or may not also reduplicate:

(49) Other uvular facts

ʔasaalχi	ʔasaalqi (dim.)	‘nape of neck’
sp’iiχʷ	sp’ip’qʷ (cont.)	‘hit the head with a stick’
qinχ	qiqnq-i (dim.)	‘shoe’
sinaχʷmiχʷ	sinaqʷmiqʷ-i (dim.)	‘ceremonial head-dress’
qʷals	qʷaaχʷls-i (dim.)	‘fir tree needles’

In earlier works, the processes that occurred only in the reduplicants themselves were not clearly distinguished from those that took place in the bases and roots associated with reduplicants. It appears that they are somewhat different, though. In reduplicants, examples are attested of almost all types of velar and uvular segments spirantizing to *x*, as exemplified in (48), but no examples of fortition have been found. In bases and roots, I have only found examples of uvular segments changing, as above in (49), and most examples have the uvular fricative changing to a uvular stop (often in roots where syncope also applies). Since this difference exists, I don’t feel that the Nuxalk evidence clearly supports a general constraint parametrization allowing velars to change continuancy values more easily than other segments. I would like to have this evidence, since constraint parametrization leads to great multiplication of the number and possible rankings of constraints, and other evidence for this particular one is not known to me. (I will not deal further with the facts in (49) about uvular roots.)

Another way to handle the velar facts here would be to claim that velars are less marked or more worthy of preservation than other types of segments. Usually, coronals are claimed to be the least marked place of articulation (or pharyngeals: Alderete et al.1996). But there are precedents for claiming that velars have some sort of precedence over other articulations. Two papers on child phonology, Gnanadesikan 1995 and Pater 1996, have noted interesting effects of sonority and articulatory place on children's outputs. They both note that a sonority restriction calling for less sonorous onsets leads to preservation of obstruents from clusters or truncated portions of words (over liquids, for example, which are too sonorous to be preferred onsets). Gnanadesikan then also notes a process of coalescence in which Labial and Dorsal features are preserved preferentially over Coronals. And Pater analyzes a harmony process in which velars are dominant over labials and coronals. The papers both capture these facts by ranking IDENT[Dorsal] over at least IDENT[Coronal], and differ in where IDENT[Labial] is ranked.

Since Nuxalk has no labial fricatives, it would be unproblematic to prevent a labial *p* from spirantizing so as to be an allowable coda, even if Labial is a fairly high-ranked feature. But actually, using IDENT[Place] constraints would not work to preserve velars in reduplicant codas, when they are in the input; that is because an IDENT[F] constraint is not violated when the correspondent segment that would bear the feature is not present, as when a velar stop in the base is not reduplicated. What would actually produce the effect seen in (48) above is a MAX[Velar]-BR constraint, which would demand preservation of the place feature in the reduplicant, and would be violated when the feature or the segment that should carry the feature is not reduplicated. This is the tack that will be taken here.[14]

4.3.2. Constraints and Analysis

After the discussion above, I would like to make concrete what this analysis can and can't capture. First of all, there are the two very robust and well-motivated coda conditions in reduplication: no stops and no secondary articulations are allowed. The prohibition on stops follows from the idea of an ideal sonority shape for syllables, which favors relatively sonorous codas. It is easy enough to ban stops from codas in reduplication, while allowing them in reduplicated onsets and the rest of the language. As long as a general markedness constraint against stops is ranked below IO-Faith, the language as a whole can contain stops; and as long as this constraint is also ranked below OnsetFaith-BR, even reduplicated onsets can be stops.

(50) *[-cont]: Stops are prohibited

The prohibition against complex segments is a classic coda condition effect. Codas aren't privileged positions, so highly marked segments can easily be excluded from them. To help capture another partially predictable pattern that will be discussed in the next section, I will rule out the complex segments through markedness constraints on each of the individual secondary articulations:

[14] A possible problem with this constraint conceptually is that MAX constraints are about preserving a feature or segment that is in the input; somehow forcing reduplication of a segment that should carry a feature, when the feature is still present in the base (= the output) already, doesn't seem like the same thing.

(51) *Phar-VPlace: Secondary pharyngeals are prohibited (as in uvular segments)
*Glot-VPlace: Secondary glottalization is prohibited
*Lab-VPlace: Secondary labialization is prohibited

These constraints rule out each of the individual secondary articulations found in Nuxalk. Segments can avoid violating these by jettisoning their secondary articulations, and keeping their primary articulations. (In the tableaux these constraints will be grouped together as *COMPSEG, for convenience.)

Finally, as mentioned above, IO-Faith and OnsetFaith must be involved, so these markedness constraints don’t affect more of the language than they are supposed to. Onsets are a salient position in the syllable, and in many languages can license more contrasts or more marked segments than the less salient Codas (Lombardi 1996, Beckman 1996). Specific faithfulness constraints for the Onset position are thus expected, to preserve contrasts, but no specific faithfulness constraint for the Coda should exist. OnsetFaith and IO-Faith will have to be ranked above the markedness constraints above, and also above general BR-Faithfulness.

First, let’s look at a root with a uvular coda segment:

(52) Uvular coda condition

/RED+ɬ’aq’^{w}/	OnsFaithBR	*COMPSEG	*[-cont]	MAX-BR	Id[cont]
a. ɬ’aq’^{w}ɬ’aq’^{w}		****!	**		
☞b. ɬ’axɬ’aq’^{w}		***	*		*
c. ɬaxɬ’aq’^{w}	*!	**	*		*
d. ɬ’aχɬ’aq’^{w}		****!	*		*
e. ɬ’aɬ’aq’^{w}		***	*	*!	

This root has many secondary articulations; IO-Faith and OnsetFaith constraints preserve them in the base and the reduplicant’s onset, so they can only be lost in the reduplicative coda. Candidate (a) preserves the base exactly, and loses on its extra COMPSEG violation. Candidate (c) fails to reduplicate the glottalization of the onset, which is fatal. Candidate (d) preserves the uvularity of the coda, which counts as a COMPSEG violation (due to the pharyngeal secondary articulation) also. Candidate (e) leaves out the coda altogether, and ties with the winning candidate all the way until the MAX constraint. The IDENT constraint must be below MAX, then, since what candidate (b) does is spirantize the coda and lose its secondary articulations, in order to preserve it; this ranking allows these changes to take place, rather than not reduplicating the coda.

This analysis appears to capture the facts for roots with uvular and velar segments. It also produces the right result for roots that have an allowable coda in the root to start with:

(53) Allowable coda

/RED+silin/	OnsFaithBR	*CompSeg	*[-cont]	Max-BR
☞a. silslin				
b. sislin				*!

With the ProsTarg(Syllable) constraint undominated, the most that can be reduplicated of this root is a closed syllable. And since nothing is wrong with that syllable in a root like this (which also undergoes syncope), reduplicating less as in (b) fatally violates Max-BR.

Now, the problem is what to do with a root that has a non-allowable coda, such as *sputx* ‘eulachon’, which reduplicates as *spuuptxi*. According to the constraint ranking above, we predict spirantization of the *t* rather than failure to reduplicate it. This is what shows that we need to change the constraint-ranking above and add a special velar constraint. The double tableau below will show how this analysis works, using Max-Velar-BR to capture the special behavior of velars and uvulars (to show the point more clearly, a hypothetical simple coronal-final form will be used):

(54) Coronal vs. Velar codas

/RED+pat/	OnsFaithBR	*CompSeg	*[-cont]	MaxVelBR	Id[cont]	MaxBR
a. patpat			****!			
b. paspat			***			*
☞c. papat			***		*!	
/RED+nik’/						
d. nik’nik’		**!	**			
☞e. nixnik’		*	*		*	
f. ninik’		*	*	*!		*

Ranking Ident[continuant] over Max makes it worse to lenite a segment than to delete it, generally; but ranking Max-Velar over Ident means that losing a velar segment is worse even than leniting a segment. So spirantizing the velar in (e) is better than keeping the complex velar stop (d) or failing to reduplicate it (f). But with a coronal stop as the potential coda, it is best to fail to reduplicate it (c), rather than leniting it (b) or keeping it (a).

So the final ranking for the Coda Condition effects is as shown above. This captures the predictability in the system: stops and complex articulations are not allowed, and velars and uvulars are allowed to spirantize to satisfy these constraints. The constraint ranking with the Peak constraints (discussed in the section 4.2. above) decides where the reduplicant will be, and thus what segments this coda ranking will be picking as possible codas. The coda patterns are not fully predictable, though, so specific ProsTarg constraints must mark some roots as only taking a CV reduplicant, regardless of what the coda would be, and restrict others to taking a CVn reduplicant.

The Coda Condition effects here can be linked to the rest of this paper by two main points. One is that they show classic emergence of the unmarked effects: onsets and roots preserve more consonantal contrasts than reduplicative codas. This is achieved by ranking markedness constraints below IO-Faith and OnsetFaith, but above general BR-Faith. The second is that the particular markedness constraints active here include a constraint against the least sonorous segmental type, stops. The ideal syllable has a slow sonority fall after the nucleus, which means that it needs a fairly sonorous coda. This restriction then is part of a general pattern of sonority restrictions on various syllabic positions, in reduplication and elsewhere.

4.4. Further Patterns

There are two more reduplicative patterns in Nuxalk that should be mentioned, although the details of their analysis are not particularly crucial to the main goals of this paper. One is the pattern taken by obstruent-only roots. These roots, while apparently comprising up to 10% of the Nuxalk lexicon (Bagemihl 1991, from Nater 1984), participate in reduplication much more rarely than would be expected. The *Peak/obstruent constraint is shown in our constraint ranking as fairly high-ranked. It could actually be ranked even higher than is shown, since it is never violated; it may be ranked above the constraints that force the reduplicant to exist at all. In that case, it would be worse to try to reduplicate an obstruent nucleus than to not reduplicate.

Now when obstruent-only stems do participate in reduplication, both the base and the reduplicant seem to have an epenthetic vowel or sonorant inserted, so the *P/obstruent constraint is not violated. I suggest that these roots are lexically marked as taking an alternate allomorph for reduplication, because otherwise there is a ranking problem. The problem is as follows. In order for epenthesis to take place in the base (allowing the reduplicant to copy it), DEP-BR and *P/obstruent need to be ranked above DEP-IO:

(55) Epenthesis

/RED+k’x/	DEP-BR	*P/obst	DEP-IO
a. k’xk’x		*!	
b. k’ixk’x	*!		
☞c. k’ixk’ix			*

With this ranking, reduplicating the root as is violates the Peak constraint (a), while epenthesizing only in the reduplicant violates DEP-BR (b), so the winner is (c). The problem with this ranking is that it is likely to produce *k’ix* as the output from the input *k’x*, with no reduplication: it could lead to back-copying. So I do not have a complete story of how this pattern works. The point is that reduplicating an obstruent nucleus is very much not preferred by this language, and two ways to avoid it are used: failure to participate in reduplication at all, and epenthesis into the root and reduplicant.

There are a few other stems which take epenthetic vowels only in their reduplicants (e.g. *smɨk*, but *sasmɨk*). Most of these stems have a sonorant consonant as the first possible nucleus for reduplication. This result could be achieved by a reranking of DEP-BR under *P/sonorant; this would make epenthesis a better option than reduplicating the sonorant consonant, for these few roots.

Finally, these is the pattern of V reduplication. This is a fairly common pattern, which seems partly predictable. The majority of the stems which take this reduplication pattern have glottalized onsets:

(57) V reduplication

k’nc	ʔnk’nc-i	‘sperm whale’
c’usm	ʔusc’usm	‘early evening’
p’wi	ʔup’wi	‘halibut’
sk’ma	smk’m-ni	‘comb’

In the section on Coda Conditions, the constraint *Glot-VPlace was postulated. One reason to separate the individual constraints against secondary articulations was to capture this pattern here. It appears that the *Glot-VPlace markedness constraint is ranked above OnsetFaith for these roots, in order to ban the glottalized onset consonant. In section 4.3., all of the markedness constraints on secondary articulations were ranked under OnsetFaith, since the secondary articulations were only banned in codas. Now we see one of the same markedness constraints actively banning reduplicant onsets.

4.5. Conclusion

The Nuxalk reduplicative system is obviously quite complicated, and there are numerous exceptional patterns and lexical exceptions to general rules. However, it seems clear that the most robust generalizations found in this data can be captured using sonority constraints. The position of the reduplicant, which varies between being a prefix and an infix, is entirely predictable from the segmental content of the root. The reduplicant will move into the root in order to find a segment that is sonorous enough to be a nucleus. The Peak constraints encode this generalization. The ALIGN-PREFIX constraint which makes the reduplicant a prefix is ranked below *P/obstruent but above *P/sonorant, so a reduplicant will infix minimally in order to find a nucleus that is at least as good as a sonorant consonant.

The coda conditions discussed in section 4.3. are also sonority-related, and appear in an emergence of the unmarked ranking schema. Codas in reduplication may not be stops: this follows from the general principle that codas should be more sonorous than onsets (Clements 1990). This constraint and the markedness constraints against secondary articulations are usually ranked below both IO-Faith and OnsetFaith, so they only affect the inventory of reduplicated codas, since codas in affixes are weak positions that don’t license as many segmental contrasts. Most of the other reduplicative patterns found in Nuxalk are at least partially explained using the constraints discussed above.

Finally, it is interesting to see how some of the core Correspondence constraints are ranked in Nuxalk, and how they interact with the sonority-related markedness constraints. Nuxalk is a case where the Alignment and Anchoring constraints are ranked quite separately: the L-ANCHOR constraint is never violated, while the Alignment constraint is often violated. CONTIGUITY, however, is highly ranked, which is the normal case in reduplication (McCarthy and Prince 1995).

5. Overall Conclusion

Nakanai (Austronesian) and Nuxalk (Salish) are widely differing languages spoken in areas thousands of miles apart. Nuxalk has many words with no vowels at all, while Nakanai rarely even has closed syllables. Yet the reduplicative patterns of both of these languages have been shown to follow from some of the same sonority-related constraints. These constraints regulate what can be the nucleus of a syllable (the Peak constraints), what can be a coda, and what a good diphthong is. These constraints all follow from the idea that there is an ideal sonority slope through a syllable (Clements 1990, Zec 1995). This ideal sonority contour begins with a sharp rise in sonority from the onset to the nucleus, and ends with a gentle or nonexistent fall in sonority from the nucleus to the coda. Thus in Nakanai and Nuxalk reduplication, relatively sonorous Peaks, relatively sonorous codas, and diphthongs that fall in sonority rather than rise are preferred. In many languages, such constraints are useful in explaining syllabification in general.

Many of these constraints show up in an emergence of the unmarked ranking schema. The markedness constraints are not visibly very active in the language as a whole, so they are ranked below IO-Faith. But by being ranked above at least one BR-Faithfulness constraint, they are allowed to affect the shape and position of reduplicants. And these languages differ in which BR-Faithfulness constraints are dominated by the sonority-related markedness constraints, leading to very different-looking outcomes. In Nakanai, the markedness constraints dominate MAX-BR and CONTIGUITY, so they compel non-reduplication and skipping of segments of the base. The Alignment and Anchoring constraints, however, are both highly ranked, so the reduplicant is always prefixed to the foot and begins with the correspondent segment of the left edge of the foot. In Nuxalk, Anchoring and CONTIGUITY are highly-ranked, but Alignment is not; thus the sonority requirements cause infixation of the normally prefixal reduplicant. Nuxalk also has a coda condition which can be linked to sonority, so an Onset-Faithfulness constraint is crucially ranked above some markedness constraints.

In sum, the sonority-related markedness constraints discussed in this paper provide most of the predictability of these partly lexical reduplication systems, in two different ways. This variety is an expected result in the OT framework, stemming from the normal reranking of universal constraints between particular languages.

References:

Alderete, John, Jill Beckman, Laura Benua, Amalia Gnanadesikan, John McCarthy, and Suzanne Urbanczyk (1996) “Reduplication and Segmental Unmarkedness.” Ms., University of Massachusetts, Amherst.

Bagemihl, Bruce (1991) “Syllable Structure in Bella Coola,” *Linguistic Inquiry* 22:589-646.

Beckman, Jill N. (1996) *Positional Faithfulness*. Ph.D. dissertation, University of Massachusetts, Amherst.

Benua, Laura (1995) “Identity Effects in Morphological Truncation,” in Jill Beckman et al., eds., *University of Massachusetts Occasional Papers in Linguistics* 18: Papers in Optimality Theory, Graduate Linguistic Student Association, Amherst.

Benua, Laura (1997) *Transderivational Identity: Phonological Relations Between Words*. Ph.D. dissertation, University of Massachusetts, Amherst.

Broselow, E. and John McCarthy (1984) “A Theory of Internal Reduplication,” *The Linguistic Review* 3: 25-88.

Clements, G.N. (1990) “The Role of the Sonority Cycle in Core Syllabification,” in J. Kingston & Mary Beckman, eds., *Papers in Laboratory Phonology I: Between the Grammar and Physics of Speech*. Cambridge University Press, Cambridge.

Dell, François and Mohamed Elmedlaoui (1985) “Syllabic Consonants and Syllabification in Imdlawn Tashlhiyt Berber,” *Journal of African Languages and Linguistics* 7: 105-130.

Gafos, Diamandis (1996) *The Articulatory Basis of Locality in Phonology*. Ph.D. dissertation, Johns Hopkins University.

Gafos, Diamandis (1997a) “Inferring A-Templatic Reduplicative Affixation: A Lexical Parameter Learnability Result.” Ms., University of Massachusetts, Amherst.

Gafos, Diamandis (1997b) “On Anchoring.” Ms., University of Massachusetts, Amherst.

Gnanadesikan, Amalia (1995) “Markedness and Faithfulness Constraints in Child Phonology.” Ms., University of Massachusetts, Amherst.

Hoard, James E. (1978) “Syllabification in Northwest Indian Languages, With Remarks on the Nature of Syllabic Stops and Affricates,” in A. Bell and J.B. Hooper, eds., *Syllables and Segments*. North-Holland Publishing Company, Amsterdam and New York.

Ito, Junko and Armin Mester (1995) “The Core-Periphery Structure of the Lexicon and Constraints on Reranking,” in Jill Beckman et al., eds., *University of Massachusetts Occasional Papers in Linguistics* 18: Papers in Optimality Theory, Graduate Linguistic Student Association, Amherst.

Johnston, Raymond L. (1980) *Nakanai of New Britain: The Grammar of an Oceanic Language*. (Pacific Linguistics Series B, no. 70) Australian National University, Canberra.

Jones, Caroline (1997) “Discontiguity in Umpila and Mangarrayi.” Ms., University of Massachusetts, Amherst.

Kaye, Jonathan (1983) “On the Syllable Structure of Certain West African Languages,” in D.L. Goyvaerts, ed., *African Linguistics--Essays in Memory of M. W. K. Semikenke*. Benjamins, Amsterdam.

Lombardi, Linda (1996) “Positional Faithfulness and the Phonology of Voicing in Optimality Theory.” Ms., University of Maryland, College Park.

McCarthy, John (1994) “The Phonetics and Phonology of Semitic Pharyngeals,” in Patricia Keating, ed., *Papers in Laboratory Phonology III: Phonological Structure and Phonetic Form,* Cambridge University Press, Cambridge.

McCarthy, John and Alan Prince (1993) *Prosodic Morphology I: Constraint Interaction and Satisfaction*. Ms., University of Massachusetts, Amherst and Rutgers University.

McCarthy, John and Alan Prince (1994) “The Emergence of the Unmarked: Optimality in Prosodic Morphology,” in Merce Gonzalez, ed., *Proceedings of the North East Linguistic Society* 24. Graduate Linguistic Student Association, University of Massachusetts, Amherst.

McCarthy, John and Alan Prince (1995) “Faithfulness and Reduplicative Identity,” in Jill Beckman et al., eds., *University of Massachusetts Occasional Papers in Linguistics* 18: Papers in Optimality Theory, Graduate Linguistic Student Association, Amherst.

Nater, H.F. (1984) *The Bella Coola Language*. National Museum of Man Mercury Series, Canadian Ethnology Service Paper No. 92. National Museums of Canada, Ottawa.

Newman, Stanley (1947) “Bella Coola I: Phonology,” *IJAL* 13: 129-134.

Newman, Stanley (1971) “Bella Coola Reduplication,” *IJAL* 37: 34-38.

Pater, Joe (1996) “Minimal Violation and Phonological Development.” Ms., McGill University and the University of British Columbia.

Prince, Alan and Paul Smolensky (1993) *Optimality Theory: Constraint Interaction in Generative Grammar*. Ms., Rutgers University, New Brunswick, and University of Colorado, Boulder. RuCCS-TR-2.

Rosenthall, Samuel (1994) *Vowel/Glide Alternation in a Theory of Constraint Interaction*. Ph.D. dissertation, University of Massachusetts, Amherst.

Shaw, Patricia (1987) “Non-conservation of Melodic Structure in Reduplication,” *CLS* 23.

Shim, Minsu (1996) “Edge Reduplication and Anchoring in Correspondence Theory,” to appear in *BLS* 22.

Smith, N.S.H. (1981) “Foley's Scales of Relative Phonological Strength,” in D.L. Goyvaerts, ed., *Phonology in the 1980's* (Story-Scientia Linguistics Series 4), E. Story-Scientia, Ghent, Belgium.

Spaelti, Philip (1997) *Dimensions of Variation in Multi-Pattern Reduplication*. Ph.D. dissertation, University of California, Santa Cruz.

Steriade, Donca (1988) “Reduplication and Syllable Transfer in Sanskrit and Elsewhere,” *Phonology* 5: 73-155.

Urbanczyk, Suzanne (1995) “Double Reduplications in Parallel,” in Jill Beckman et al., eds., *University of Massachusetts Occasional Papers in Linguistics* 18: Papers in Optimality Theory, Graduate Linguistic Student Association, Amherst.

Urbanczyk, Suzanne (1996) *Patterns of Reduplication in Lushootseed*. Ph.D. dissertation, University of Massachusetts, Amherst.

Williams, Jeffrey P. (1984) “Phonological Conditioning in Nakanai Reduplication,” *Linguistic Analysis* 14:173-190.

Zec, Draga (1995) “Sonority Constraints on Syllable Structure,” *Phonology* 12: 85-129.

Department of Linguistics
South College
University of Massachusetts
Amherst, MA 01003

katyc@linguist.umass.edu