Chapter 2 Typology of OCP on Features in OT

2.1 OCP in Previous Research

This chapter will discuss a typological study of the OCP on features. In this section, I will review the previous research on OCP. First, in section 2.1, I will examine the OCP in autosegmental phonology, and point out the problems raised within that framework. Next, I will discuss the OCP in Optimality Theory in section 2.2. In this section, I will make clear how the OCP is considered in OT, how the problems raised in autosegmental phonology can be solved in OT, and how I treat the OCP within the framework of OT.

2.1.1 OCP in Autosegmental Phonology

2.1.1.1 Leben (1973) and Goldsmith (1976)

Leben (1973) first proposed a principle to forbid two identical tones from being adjacent at the melodic level in the analysis of the Tiv Verbal system. He claimed that a sequence on the tones such as HHL is impossible in the language because the principle rules it out.

Goldsmith (1976) calls this principle the "Obligatory Contour Principle (OCP)", and states it as follows:

(1) Obligatory Contour Principle (Leben) (from Goldsmith 1976:36):

> At the melodic level of the grammar, any two adjacent tonemes must
> be distinct. Thus HHL is not a possible melodic pattern; it
> automatically simplifies to HL.

Goldsmith is skeptical about the existence of such a principle in UG, because not all tone languages express the same simplification of HHL into HL. This begins the debate on the inclusion or rejection of the OCP in the theory.

2.1.1.2 McCarthy (1986), Myers (1987) and Odden (1986, 1988)

The OCP is originally the principle used to prevent underlying representations from containing identical adjacent elements on autosegmental tiers. For instance, the principle forces the following multiple linking of the tone to avoid violating the OCP:

(2) Multiply linked high tone in Shona (Myers 1987, Odden 1980):

```
                hákáta  'diviner's bones'

        *(a) hakata              (b) hakata
               |  |                    \  /
              H  H                      H
```

The underlying representation is presumed (b) due to the OCP.

Although the original principle bars only identical adjacent autosegments in underlying representations, several researchers have tried to extend it into a more general principle which applies at the level of segments as well as autosegments. McCarthy (1986) indicates that the principle plays the role of a rule-blocker. He claims that there are many languages in which a rule of syncope is blocked when the rule would violate the OCP (antigemination effect). Following McCarthy's paradigm, Yip (1988) proposes that the OCP also works as a rule-trigger. She notes various repair strategies to avoid OCP violations, such as degemination (deletion), dissimilation, assimilation, and epenthesis. The rules that give rise to these processes are triggered by the OCP.

In contrast to those works in which the OCP is extended to a more general principle, arguments from the opposite viewpoint have also been submitted. Odden (1986, 1988) criticizes McCarthy's antigemination analysis, because not all languages show avoidance of identical elements. According to Odden (1986, 1988), this inconsistency could not be explained if the OCP were a principle of UG.

2.1.1.3 Summary

The status of OCP has thus been debated in autosegmental phonology, and as Goldsmith (1990) concludes, since the OCP is a "soft" universal principle, the problem of the universality of OCP is not resolved within the framework.

2.1.2 OCP in OT

The matter of the universality of the OCP constraint as a property of UG, so central to autosegmental phonology, is not problematic within the OT framework. The reason is that OT allows for constraint violation, therefore the OCP will be expected to be active when it is ranked highly enough to be respected, while it will be violated when it is ranked lower than some other constraints that must be satisfied.

However, another question arises. As already noted in section 1.2, is the OCP a primitive component of UG by itself as widely assumed, or do the properties of UG give rise to the effect of the OCP (Itô and Mester 1996, Alderete 1997)? Following Itô and Mester (1996) and Alderete 1997, I adopt the idea that the OCP is derived from self-conjunction of markedness. However, I use OCP[F] to refer to the self-conjoined markedness constraints, i.e.*[$_{\text{DOMAIN}}$ [F][F]] for typographical convenience, and also it does not deny the possibility of the OCP as the primitive constraint. I leave the issue of the absolute nature of the OCP for future research.

Although I also assume that the domain for the OCP constraint is specified language-specifically in this dissertation, I will consider whether and how the domain is specified in each language from the viewpoint of language learnability in future investigation.

This section recapitulates the work of Itô and Mester (1996) and Alderete (1997) which introduce the idea that the OCP is a self-conjunction of markedness constraints in certain local domains.

2.1.2.1 OCP as Self-conjunction of Markedness Constraints (Itô and Mester 1996, Alderete 1997)

Itô and Mester (1996) and Alderete (1997) independently propose that the OCP is a self-conjunction of markedness constraints in some local domain. This idea is formulated in the mold of Local Conjunction (Smolensky 1993, 1995, 1997).

Local Conjunction obtains when two lower-ranked constraints are conjoined in a certain local domain to play the role of one higher-ranked constraint.[1]

Suppose that constraint A and Constraint B are both relatively low-ranked in language C so that both are violated to satisfy the higher ranked constraint D. However, when A and B are conjoined, it is assumed that A&B could be ranked above D. Then, A&B must be respected at the expense of a violation of D.

(3) A&B >> D >> A, B

Thus, the ranking given in (3) is established in language C.

1 A detailed examination of Local Conjunction is given in chapter 4.

This concept of Local Conjunction is used to formalize OCP constraints as self-conjoined markedness constraints. In language E, a markedness constraint *F, which prohibits the marked element F, is lower-ranked than another markedness constraint *G, which forbids the marked element G. This ranking lets F surface in some data of the language unless some faithfulness constraint intervenes, because F is more unmarked than G.

However, in some local domain H within the same language E, it is also possible that the sequence of FF is banned, when *F is conjoined with *F into a self-conjoined constraint in the domain H: $[*F*F]_H$, and the self-conjoined constraint $[*F*F]_H$ is higher ranked than *G.

(4) $[*F*F]_H >> *G >> *F$

Itô and Mester (1996), and Alderete (1997) thus reach the conclusion that a constraint specifically for OCP is redundant because the full set of the self-conjoined markedness constraints can account for all OCP phenomena.

Their approach has broader empirical coverage than a simple OCP constraint. First, with a simple OCP constraint it is difficult to account for the variation in adjacency (so-called "long distance OCP") (Archangeli and Pullyblank 1994, Odden 1994, Suzuki 1998).

This problem is resolved in the self-conjoined markedness constraint model of the OCP. The first problem is solved by defining the local domain for each local conjunction (Smolensky 1993, 1995, 1997). For instance, Itô and Mester (1996), and Alderete (1997) propose a self-conjunction, $*[+voice, –sonorant]^2_{stem}$: two voiced obstruents are not allowed per stem, to account for Rendaku (Sequential

Voicing) and Lyman's Law in Japanese. When two words compound into one in Japanese, the initial obstruent of the second compound member becomes voiced. This is called Rendaku (Sequential voicing). However, Rendaku does not take place if the second member of the compound already contains a voiced obstruent within its morpheme due to Lyman's Law which prohibits more than one voiced obstruent in a morpheme. Thus, in order to ban the occurrence of more than one voiced obstruent within the second member of the compound (within a stem), a self-conjoined constraint is introduced.

Second, with only a simple OCP, it is hard to explain OCP effects on elements other than those on the same autosegmental tier. For instance, vowel length is not a single autosegmental element. Therefore, it is problematic in the previous framework how to account for the prohibition of two longs vowels in a certain domain. This problem is solved by one of Alderete's (1997) self-conjunctions. In his analysis of vowel length dissimilation in Oromo, he proposes a self-conjunction, *NoLongVowel$^2_{SA}$: In adjacent syllables, avoid two vowels each dominated by more than one mora. With this constraint, the fact that more than one long vowel is prevented from occurring in adjacent syllables (neither segmental nor tier adjacency is relevant) is accounted for.

Suzuki (1998) argues that the self-conjunction approach to the OCP is too restrictive because it considers that the OCP prohibits the cooccurrence of the marked structure. The existence of the single markedness constraint is necessary to bear the self-conjoined constraint, and it cannot account for the OCP effects on unmarked features.

However, I argue against Suzuki's claim, because there is no definite unmarked feature when we consider features within the framework of Optimality

Theory. All the features are marked in a sense; therefore, the constraints which prohibit them exist in the grammar, and some of the features are relatively unmarked compared to the other features based on the ranking of those markedness constraints. Thus, for example, although the [cor] feature is relatively unmarked compared to the [dor] or [lab] feature, it is still marked, and the constraint against the [cor] feature does exist, namely, *[cor]. The OCP effects on this relatively unmarked feature [cor] can be accounted for based on the self-conjoined markedness constraint, namely, *[cor][cor]. In this respect, there are no OCP effects which the self-conjunction approach cannot explain.[2]

One thing which I argue against is "*Theorem of coronal unmarkedness in segmental cooccurrence restrictions:*" proposed by Alderete (1997) to claim that a segmental cooccurrence restriction on [coronal] entails the same cooccurrence restriction for [labial] and [dorsal] (p.29).

The status of local conjunction as a property of UG is still controversial. As Fukazawa and Miglio (1996, to appear), and Miglio and Fukazawa (1997) claim, only the potential of conjunction, namely, the "&-operator" for Local Conjunction, is a property of UG. Each local conjunction constraint is language-specific. If the existence of each local conjunction constraint is language-specific, then a segmental cooccurrence restriction on [coronal] does not have to entail the same cooccurrence restriction [dorsal] (or [labial]). Such an entailment must be held only when a certain

2 There seems to be a potential problem of the existence of the self-conjoined constraint of the minus value of some feature. However, as I already mentioned in section 1.2, I assume featural privativity; therefore, there is no minus value for features. All features are fully specified. For instance, there is a markedness constraint for the feature *[voice]; hence, there is a self-conjoined OCP constraint, *[voice][voice]. On the other hand, there is no minus value for this feature such as [–voice]. Thus, there is not a constraint *[–voice][–voice]. Dahl's Law in Bantu has been analyzed this way, but see Lombardi (1995c) for an alternative analysis.

language contains both *Coronal$^{2}_{Domain}$ and *Dorsal$^{2}_{Domain}$ (or *Labial$^{2}_{Domain}$). As Smolensky (1995) proposes, the ranking *Dorsal$^{2}_{Domain}$, *Labial$^{2}_{Domain}$ >> Coronal$^{2}_{Domain}$ is universal, due to the universal of the hierarchy of the markedness constraints for the place features as introduced in (5).

(5) Universal Ranking for the markedness constraints for place features

(Prince and Smolensky 1993)

*Labial, *Dorsal >> *Coronal

However, it is possible that a language lacks some of the self-conjoined markedness constraint, such as *Dorsal$^{2}_{Domain}$, and carries only Coronal$^{2}_{Domain}$. In that case, the ranking *Dorsal$^{2}_{Domain}$, *Labial$^{2}_{Domain}$ >> Coronal$^{2}_{Domain}$ is not possible in the language.

Suzuki cites Alderete's (1997) theorem to criticize the self-conjunction approach to OCP by arguing that this theorem does not hold in languages such as Dakota or Akan, where only a cooccurrence restriction on [coronal] is observed, and no restriction on the more marked [lab] + [dor] is observed.

However, as I already point out, if local conjunction is language-specific. Therefore, Alderete's theorem does not hold, and there is nothing problematic in the self-conjunction approach to OCP.

For example, in Dakota, the sequence of [cor][cor] turns into [dor][cor]. This is the very case illustrated in (4) above. There is a universal ranking of the marked constraints for place features proposed by Prince and Smolensky (1993)[3] as introduced above in (5).

According to this ranking, the alternation of [coronal] (less marked feature) into [dorsal] (more marked feature) looks theoretically odd. The single constraint *Dorsal is surely ranked above the single constraint *Coronal. However, a self-conjunction of *Coronal outranks single *Dorsal, resulting in the ranking: $\text{*Coronal}^2_{\text{Domain}}$ >> *Dorsal >> *Coronal. With this ranking, the case of Dakota[4] is correctly accounted for. The single coronal segment is less marked than a single dorsal segment , and a single dorsal segment is less marked than two cooccurring coronal segments within a local domain. This is illustrated in the following tableau:

(6)

/cor cor/	*[cor][cor]	*[dor]	*[cor]
a. [cor][cor]	*!		**
☞ b. [dor][cor]		*	*

Thus, I conclude that the approach of the OCP as self-conjunction can still deal with the OCP effects within the OT framework.

3 Lombardi (1995b) revises this ranking by adding another place feature [Pharyngeal], following McCarthy (1989, 1994a). Although I do not introduce this ranking in this chapter, I will discuss it in section 3.2.2.5, and in 4.5.1 in detail.

4 Dakota coronal dissimilation is analyzed in detail in section 3.2.

2.1.2.2 Summary

As I have observed in this section, the question asked in section 1 has been one of the central issues when we consider the effect of the OCP in OT: whether the OCP is a component of UG by itself, or whether the properties of UG give rise to the effect of the OCP. I claim that OCP is derived as the effect of self-conjunction, following Itô and Mester (1996), and Alderete (1997). However, my claim does not deny the possibility of OCP as the primitive component of UG. Whether OCP constraints are derived or primitive, and however they may be derived, it is clear that the result is that UG contains constraints prohibiting two identical or similar elements from appearing in the same domain. My focus will be on how such constraints affecting features interact cross-linguistically to produce the observed typology of effects. Although I assume OCP is self-conjoined markedness constraints, I use OCP[F] to refer to *[F][F] in this dissertation. I will leave the issue of primitivity of OCP for future investigation.

2.2 Typology of OCP on Features

2.2.1 Four Types of Languages

In this section, I consider the factorial typology with respect to OCP effects on features in OT. In OT, typology accounts for language universality based on the same set of the relevant constraints, and for language particularity based on the different rankings of those constraints. I will consider what kinds of constraints are necessary in the analysis of OCP effects, and how the different constraint interactions represent different results in each type of language: whether or not the OCP constraints can be violated, and what kinds of repair strategies are observed. Thus, I assert that the typology of OCP effects on features is achieved on the basis of the investigation of how the OCP constraints and markedness constraints interact with faithfulness constraints.

I will provide a classification of languages based on OCP effects on features drawing on the proposal of Yip (1988). According to Yip's claim, there are logically five types of languages in terms of OCP on features. Although I will build on Yip's claim to classify languages, I will reexamine her proposal because my classification is specifically for OCP on features. I will eliminate some of the possibilities from Yip's classification such as epenthesis of a segment, and reorganize some of the categories such as dissimilation, assimilation, and deletion. I will conclude that there are four types of languages regarding OCP on features.

In the framework of autosegmental phonology, following McCarthy's (1986) paradigm, Yip proposes that the OCP sometimes plays the role of a rule trigger. Yip (1988: 73-74) indicates that there are four kinds of repair strategies to avoid OCP violations, namely degemination, dissimilation, assimilation, and epenthesis. Therefore, it seems that there are five types of languages from the perspective of the OCP effect:

(7) A Hypothetical Typology for the OCP effects:

Type 1: OCP violation is allowed

Type 2: OCP violation is not allowed, and degemination (deletion of one segment) takes place.

Type 3: OCP violation is not allowed, and dissimilation is observed.

Type 4: OCP violation is not allowed, and assimilation occurs.

Type 5: OCP violation is not allowed, and epenthesis takes place.

Yip claims that this categorization of repair strategies holds at the segmental and autosegmental levels.

I will claim that the above classification of the languages above must be reconsidered when we examine the typology of the OCP effects specifically on features. As I already pointed out in section 1.2, OCP on features must be considered different from OCP on segments, especially when we examine the possible repair strategies.

First, it is still controversial whether epenthesis plays a role as a repair for OCP effects on features. As in the Classic Arabic case which McCarthy (1986) analyzes, each feature is arrayed on a separate tier.

(8) The ill-formed root */dbt/ (from Clements and Hume 1995: 262)

```
     d          b         t
     |          |         |
   Place     Place     Place
     |          |         |
     |      [labial]      |
 [coronal]            [coronal]
```

In (8), since the [coronal] and [labial] tiers are not ordered, or independent from each other , the two coronals are adjacent, resulting in an OCP violation. Epenthesis of a segment in this case would not avoid the OCP violation on features.

Previous research has suggested that schwa is inserted between the two coronals in the plural or the past-tense formations in English as a result of the OCP effect on [coronal] (Yip 1987, 1989, Borowsky 1987), e.g. 'want'–'wanted(past)' ([want]–[wantəd]), 'miss'–'misses (plural)' ([mIs]–[mIsəz]).

However, if tiers are independent, it is not plausible that epenthesis is effective in avoiding an OCP violation on [coronal], because the two tiers can be still adjacent despite the fact that the segments are not adjacent:

(9)

(a) [want ə d]
        | |
   [cor] [cor]

(b) [mIs ə z]
        | |
   [cor] [cor]

Let us analyze this type of example in an actual tableau.

(10)

/t. d/ | | [cor] [cor]	OCP[coronal]	DEP-IO
☞a. /t. d/ | | [cor] [cor]	*	
b. /t ə d/ | | [cor] [cor]	*	*!

As illustrated in tableau (10), candidate (b), with epenthesis of schwa, still violates OCP[coronal]. Moreover, it violates an additional constraint, DEP-IO: all the output segments have the correspondent in the input (McCarthy and Prince 1995).

Furthermore, many English words even include two adjacent segments which bearing the feature [coronal]:

(11)

n e s t
      | |
  [cor] [cor]

Words such as (11) obviously show that a constraint for OCP on [coronal] should not be a high-ranked constraint in English. Thus, some other reason must force the schwa epenthesis in the English past-tense or plural suffix. The problem which triggers epenthesis is not only the OCP violation of [cor][cor] in this English case.[5] Therefore, this case does not constitute evidence that epenthesis can repair OCP violations on features.

There would be one possible way for epenthesis to repair OCP on features as I have already referred in section 1.2. That is when the epenthetic segment bears the opposite value of the adjacent identical feature.

(12)

```
  X    Y    Z
  |    |    |
[+F] [–F] [+F]
```

In the hypothetical example in (12), the two identical features [+F][+F] are not adjacent anymore when the segment Y which bears the opposite value for the same feature is epenthesized. However, I assume that all the features are privative.

---

5 I will not further discuss the schwa epenthesis in English, because the topic is not relevant to the theme of the dissertation. In order to account for this epenthesis, we have to examine other elements aside from the OCP on [cor]. If these elements are also other OCP effects on features, they must involve additional features. For example, we have to consider the manner feature to account for this of English epenthesis. For example, schwa insertion is observed in the past tense of the word ending {t, d}, and also in the plural form of the word ending {s, z}. However, schwa insertion is not observed in the past tense of the word ending {s, z}, e.g. the past tense of 'seize' /si:z/ is not [si:zəd] but [si:zd]. Thus, not only the coronal feature but also the manner feature is also important. Also, schwa insertion is not observed for every segment which bears coronal, only alveolars. I will not address this further but see chapter 4 for an analysis of a multiple feature OCP. This case may also crucially involve restricting the domain of the OCP to a particular type of cluster: I leave this for future research.

Therefore, there is neither [+F] nor [–F], just [F]. Thus, I conclude that there is no possibility for the epenthesis of a segment to repair the effect of OCP on features. Therefore, I eliminate epenthesis from my typology of OCP on features, and will not discuss it further in this dissertation.

Secondly, degemination, dissimilation and assimilation must be classified as repair strategies for OCP effects on features. There are several types of degemination, dissimilation and/or assimilation repair strategies: deletion of a feature, deletion of a feature with segmental deletion, deletion of a feature with featural insertion, featural fusion, and so on.

Therefore, I propose that the typology of the OCP effects on features can be explained on the basis of the following four types.

(13) Typology of the OCP effects on features:

(a) **Type 1 language: OCP violation is observed.**

(b) **Type 2 language: OCP violation is not allowed, and Featural Fusion takes place (Dissimilation & Assimilation)**

(c) **Type 3 language: OCP violation is not allowed, and Feature Deletion and Feature Insertion both occur. (Dissimilation)**

(d) **Type 4 language: OCP violation is not allowed, and Feature Deletion leads to Segmental Deletion. (Deletion)**

In the following sections, first I will illustrate each type in more detail. Next, on the basis of this classification, I will actually observe how the OCP and markedness constraints interact with faithfulness constraints in each type.

2.2.1.1 Type 1: OCP Violation

In a Type 1 language, the OCP violation for some feature is observed as the following example shows:

(14) A hypothetical example for type 1:

```
/t    o     t/ -> [ t   o   t]
 |          |         |       |
[cor][cor]     [cor][cor]
```

There are many languages which allow OCP violation for various features. In English, [tot] is a well-formed sequence despite an OCP[coronal] violation.

2.2.1.2 Type 2: Feature Fusion

In this type, features are fused to avoid a violation of the OCP on features.

(15) A hypothetical example for type 2:

```
/t    o     t/ -> [ t   o   t]
 |          |          \    /
[cor][cor]          [cor]
```

Many languages whose grammars contain Morpheme Structure Constraints (MSC) belong to this type. For example, in Ponapean, two [labial] features which do not share the same value for backness cannot be adjacent at the tier level, while those which share the same backness can be next to each other.

(16) Ponapean Labials (Mester 1986):

p + p	paip	'boulder'
	pap	'swim'
m + m	mem	'sweet'
	kamam	'to enjoy kava'
p + m	parem	'nipa palm'
	madep	'species of sea cucumber'
p^w + p^w	p^wup^w 'to fall'	
	p^wop^we	'shoulder'
m^w + m^w	sum^wum^w	'trouchus'
	kam^wam^w	'to exhaust'
	m^waam^w	'fish'
m^w + p^w	m^wop^w	'out of breath'
*p^w ap	*p^w ap	DOES NOT EXIST

Mester (1986) accounts for these Ponapean domain using the OCP. p^w indicates not labialized, but velarized. Therefore, [p] does not bear the [back] feature, while velarized [p^w] bears [back]. Mester claims that the [back] feature is dependent on the [lab] feature tier. When two [labial] features share the same value for backness, they can be fused into one labial. Hence, the OCP violation on [labial] does not take place. On the other hand, when the two [labial] features do not share the same value for the [back] feature, they cannot be fused, resulting in an OCP violation on [labial]. Consequently, two [labial] features with different values for backness will never surface in this language.

(17) Possible and Impossible sequences in Ponapean:

```
(a).   [back]        (b).               *(c).  [back]
         |                                       |
       [lab]            [lab]                  [lab]    [lab]
        /  \            /   \                    |        |
       C a C           C a C                     C   a    C
       pw a pw          p  a p                   pw  a    p
      (possible)      (possible)               (impossible)
```

Thus, the sequences in (17a) and (17b) are possible, while a sequence like (17c) is never observed in Ponapean. I will analyze these phenomena in Ponapean in detail in section 3.1.

Mester (1986) examines several other similar cases. For instance, in Alur, the alveolar stops [t] and [d], and the interdental stops [θ] and [ð] are mutually exclusive in CVC roots. In many languages, vowel harmony is observed such as in Ngbaka (Mester 1986), Ainu (Mester 1986, Itô 1984), Yawelmani Yokuts (Mester 1986 etc.), Turkish (Mester 1986), and Kirghiz (rounding harmony: Mester 1986). When two vowels become identical as a result of vowel harmony, they can be fused, resulting in the satisfaction of the OCP. According to Mester's claim, two adjacent features fuse so as not to violate the OCP in all these languages. I conclude that all these languages belong to Type 2.

2.2.1.3 Type 3: Feature Deletion and Insertion

In this type, one of the two adjacent features is deleted to avoid an OCP violation. Moreover, a new feature is inserted in order to avoid a subsequent ill-formed structure.

(18) A hypothetical example for Type 3:

```
                    [lab]
                      |
/t   o    t/ -> [ t  o  p]
 |        |       |
[cor][cor]   [cor]
```

In the example in (18), a new feature [lab] is inserted. Otherwise, the segment would be placeless due to deletion of the [coronal] feature. Building on Padgett's (1994, 1995a) proposal of a constraint against placeless segments, I assume that placeless segments do not surface in this type of language, because of his constraint, HAVEPLACE.

Suzuki (1998) examines previous research and identifies a number of languages in which both feature deletion and insertion are observed. Among the cases which Suzuki identifies, I believe that the following cases belong to Type 3:

Table I: Example Languages in Type 3:

Language	Phenomenon	References
(a) Latin	/l.l/ → [r.l]	Kent 1945:153, Odden 1994:314, Steriade 1987, 1995:154, Walsh-Dicky 1997:159
(b) Akkadian	/m.lab/ → [n.lab]	Hume 1992:113, Odden 1994:321
(c) Cantonese (a secret language)	/lab.lab/ → [lab.cor]	Yip 1982:657, 1988:83, Hume 1992:111
(d) Dakota	/cor.cor/ → [dor.cor]	Shaw 1980, 1985:184
(e) Kuman	/l.l/ → [l.r]	Walsh-Dicky 1997:155
(f) Yimas	/r.r/ → [r.t]	Forely 1991:56, Odden 1994: 316, Walsh-Dicky 1997:155
(g) Georgian	/r.r/ → [r.l]	Odden 1994:314, Walsh-Dicky 1997:155-156
(h) Tashlhiyt Berber	/m.lab/ → [n.lab]	Odden 1994:319
(i) Yidin	/l.l/ → [r.l]	Dixon 1977:99, Steriade 1995:154, Walsh-Dicky 1997:161
(j) Ainu	/r.r/ → [n.r]	Shibatani 1990:13
(k) Modern Greek	/r.r/ → [l.r]	Walsh-Dicky 1997:155

In these languages, new features are inserted to avoid the ill-formed structure brought about by deletion of a feature.

Let us examine some of the examples in Table I. In Akkadian, two labial features are adjacent. Therefore, in order to avoid the violation of OCP[lab], the first lab feature deletes, resulting in a placeless segment. Then, a relatively unmarked place feature is inserted so as not to violate HAVEPLACE which prohibits a placeless segment.

(19) (b) Akkadian: /m. lab/→ [n.lab]

```
  /m.  X/
    |    |
[lab]  [lab]

       ⇓

[cor]
    |
  [n,    X]
    ‡      |
 [lab]  [lab]
```

Cantonese and Tashlhiyt Berber are accounted for in a similar way to Akkadian.

Similarly, in the case of Dakota, there are two coronals adjacent. Therefore, one of the coronal feature deletes in order not to violate OCP[cor], and the dorsal place feature is inserted so as not to violate HAVEPLACE.[6]

All the other cases in Table II are related to liquids. Walsh-Dicky (1997) argued against previous research such as Steriade (1987, 1995) etc. which analyzed the OCP effect on liquids from the aspects of the plus and the minus values.

---

6 As for the reason why the inserted place feature is dorsal, see the detailed analysis of Dakota later in section 3.2.2.

Let us take a look at an actual example. In the case of Latin, Steriade (1987, 1995) analyzed that the alternation of /l/ into [r] in the sequence as the change of [+lateral] into [–lateral] so as not to violate OCP[+lateral].

Walsh-Dicky argued against Steriade's claim from the viewpoint of feature privativity. She claimed that there is no feature such as [lateral], and proposed a new structure of liquids. All the liquids possess the feature [liquid] which is depended on by the place feature as the first articulation. Another place feature, the secondary articulation depends on the place feature of the first articulation. Therefore, liquids are formulated as follows:

(20)

```
  R
  |
[liquid]
  |
[Place/first articulation]
  |
[Place/secondary articulation]
```

Laterals such as [l] are defined as liquids with a coronal first articulation, and with a dorsal secondary articulation. On the other hand, Rhotics such as [r] are defined as liquids with a coronal first articulation, and with a coronal (apical and/or laminal) secondary articulation.

Thus, the sequence of /l. l/ bear two adjacent dorsal place features. The place feature of the secondary articulation [dor] deletes so as not to violate OCP[dor: secondary articulation)], resulting in lack of a secondary articulation. Some constraint prohibits liquids without a secondary articulation node. Hence, the default place feature is inserted illustrated:

(21)

```
            /l   .   l/
              |       |
         [liquid][liquid]
              |       |
          [cor]  [cor]
            |        |
           [dor]  [dor]

                ⇓

             [r    .   l]
               |         |
          [liquid][liquid]
              |          |
          [cor]     [cor]
         /    ‡          |
        /  [dor]     [dor]
 [cor]
```

In the other cases of liquids, a similar explanation can be applied to. Thus, in Type 3, some privative feature deletes so as not to violate the OCP on that feature, and another feature is inserted in order to avoid the violation of the constraint which prohibits some ill-formed structure.

2.2.1.4 Type 4: Featural and Segmental Deletion

When a feature deletes, the segment on which the feature depends sometimes deletes, too. This is another kind of repair strategy which not only avoids an OCP violation, but also avoids ill-formedness due to the lack of a feature.

(22) A hypothetical example for type 4:

```
  /t   o    t/ -> [ t  o   ]
   |         |        |
[cor][cor]   [cor]
```

In this type, when the feature deletes from the segment, it creates the ill-formed structure similar to that seen in the Type 3 language above. In Type 3, however, the ill-formedness is rescued by insertion of another feature, while the entire ill-formed segment deletes in Type 4.

In Basque, when two stops are adjacent, the second [stop] feature deletes, resulting in the entire segment deleting (Hualde 1987, 1988, 1991, Lombardi 1990a, b). In Catalan, when two adjacent segments share the same place features, one of the features deletes, resulting in entire segment deletion (Morales 1992). Seri exhibits segmental deletion as the result of deleting an adjacent [constricted glottis] feature (Marlett and Stemberger 1983:628, Yip 1988:75). I conclude that these cases belong to Type 4 in which both featural and segmental deletions are observed.

The next section will discuss how these four types are explained on the basis of the difference of constraint rankings.

2.3 The Difference of the Constraint Ranking Representing Each Type in Typology

2.3.1 The Constraints for OCP on Features

Now, I will examine the kinds of constraints which account for the typology. I will observe how the OCP and markedness constraints interact with faithfulness constraints. First, I will discuss the constraints for OCP on features. As discussed in section 2.1.2.1, I assume that OCP as self-conjoined markedness constraints. However, for the sake of discussion throughout this dissertation, I will use "OCP[F]" to refer to the constraint in my analysis:

(23) OCP[F]:

Adjacent identical features are prohibited (Leben 1973; Goldsmith 1976; Mester 1986; McCarthy 1986)

When this OCP constraint outranks the relevant faithfulness constraints, some alternations are forced to take place so that the OCP constraints can be satisfied.

(24) Alternation observed:

OCP[F] >> faithfulness constraints

Type 2, 3 and 4 languages discussed above demonstrate such a constraint ranking, because repair strategies are observed to satisfy the OCPs in those types of languages. The different strategies used to repair OCP violations among these three types is determined according to the type, and ranking, of faithfulness constraints. The detailed interactions of faithfulness constraints will be examined in the next section.

On the other hand, when the faithfulness constraints are higher ranked than the OCP, the surface form will contain some OCP violation.

(25) OCP violation observed:

all the relevant faithfulness constraint(s) >> OCP[F]

This constraint ranking is observed in Type 1 languages. The OCP constraint can be violated to let the candidate which is faithful to the input win.

2.3.2 The Faithfulness Constraints for Features

This section will discuss what kind of faithfulness constraints are necessary to show the OCP effects on features. As already illustrated in section 2.3.1, OCP[F] must outrank some faithfulness constraints to produce alternations as repair strategies to avoid OCP violations in types 2, 3, and 4 languages.

However, there is a full set of faithfulness constraints for each type of faithfulness relation or for each linguistic string in Universal Grammar according to McCarthy and Prince's (1995) Correspondence Theory. These faithfulness constraints are not ranked together, but rather some of them are higher ranked than others or than the markedness constraints depending on the particular grammar. The difference in the ranking of faithfulness constraints, therefore, will define the difference in the repair strategies among types 2, 3, and 4.

The next section first reviews Correspondence Theory (McCarthy and Prince 1995), and considers how we can apply this theory to account for the typology.

2.3.2.1 Correspondence Theory (McCarthy and Prince 1995)

Correspondence Theory (McCarthy and Prince 1995) revises the original approach to input-output faithfulness in Optimality Theory (Prince and Smolensky 1993):

(26) Correspondence (McCarthy and Prince (1995), pp. 262))

Given two strings S_1 and S_2, **correspondence** is a relation **R** from the elements of S_1 to those of S_2. Elements $\alpha \in S_1$ and $\beta \in S_2$ are referred to as **correspondents** of one another when α **R** β.

All correspondence relations are generalized under this definition. Attention is given to correspondence between representations, and faithfulness constraints are itemized from the segmental, featural, or structural viewpoint: {MAX, DEP, IDENT[F], CONTIGUITY, LINEARITY, INTEGRITY, UNIFORMITY, ANCHOR, ALIGN}.

(27) Definitions for each corresponding constraint (McCarthy and Prince 1995: 370-372):

(a) MAX: Every element of S_1 has a correspondent in S_2.
Domain (**R**) = S_1

(b) DEP: Every element of S_2 has a correspondent in S_1.
Range (**R**) = S_2

(c) IDENT[F]: Correspondent segments have identical values for the feature F.
If x**R** y and x is [γF], then y is [γF].

(d) CONTIGUITY:

(d-1) I-CONTIGUITY ("No Skipping")
The portion of S_1 standing in correspondence forms a contiguous string.
Domain (**R**) is a single contiguous string in S_1.

(d-2) O-CONTIGUITY ("No Intrusion")
The portion of S_2 standing in correspondence forms a contiguous string.
Range (**R**) is a single contiguous string in S_2.

(e) {RIGHT, LEFT}-ANCHOR(S_1,S_2): Any element at the designated periphery of S_1 has a correspondent at the designated periphery of S_2.

(f) LINEARITY ("No Metathesis"): S_1 is consistent with the precedence structure of S_2, and vice versa.

(g) UNIFORMITY ("No Coalescence"): No element of S_2 has multiple correspondents in S_1.

(h) INTEGRITY ("No Breaking"): No element of S_1 has multiple correspondents in S_2.

Moreover, Correspondence Theory recognizes identity between distinct types of the representations such as Input-Output (IO), Output-Output (OO), Base-Reduplicant (BR), Tone-Tone-bearer (TT), etc. The definition of correspondence in (26) subsumes all types of linguistic relations, and every relation generates a full set of faithfulness constraints: IO:{MAX-IO, DEP-IO, IDENT[F]-IO, INTEGRITY-IO, ...}; OO:{MAX-OO, DEP-OO, IDENT[F]-OO,...}; BR:{MAX-BR, DEP-BR, ...}; etc.

Although the full set of faithfulness constraints can be instantiated for every linguistic string as indicated above, the faithfulness constraints proposed in McCarthy and Prince (1995) target the corresponding relationships only at the segmental level, and featural identity is evaluated only from the perspective of the segment, i.e. IDENT[F].

Table II: Correspondence Constraints for Segments

Type of Constraint	prohibits
MAX	Segmental Deletion
DEP	Segmental Epenthesis
IDENT[F]	Featural Change w.r.t. the segments
(I,O)-CONTIGUITY	Segmental Skipping (I-CONTIG) Segmental Intrusion (O-CONTIG)
ANCHOR ALIGNMENT	Segmental Misalignment
LINEARITY	Segmental Metathesis
UNIFORMITY	Segmental Coalescence
INTEGRITY	Segmental Splitting

Thus, each phonological alternation such as deletion, epenthesis, skipping, and so on at the segment level is constrained by the distinct type of correspondence constraint above.

However, the only faithfulness constraint on features is IDENT[F]. This constraint cannot restrict featural faithfulness when the segment upon which the feature depends deletes. For instance, consider the following hypothetical tableau.

(28) Hypothetical Tableau

	/tot/	IDENT[stop]
a.	[to]	√
b.	[tot]	√

In tableau (28), a segment deletes in candidate (a), while no alternation is observed in candidate (b). Although not only the segment but also the [stop] feature delete in candidate (a), the featural faithfulness constraint IDENT[stop] is vacuously satisfied. This is because IDENT[[stop] is relevant only when a corresponding segment exists. Consequently, there is no difference between candidate (a) and (b) in terms of IDENT[stop].

2.3.2.2 Featural Faithfulness in Optimality Theory

As we have observed in section 2.3.2.1, faithfulness constraints proposed in Correspondence Theory cannot (or can only in a limited way) independently account for featural identity. To determine which faithfulness constraints are necessary to account for the typology, we should first consider how to treat features with respect to faithfulness constraints, and how to set the faithfulness constraints for features.

As I have already stated, when features are treated as independent elements of segments, a full set of faithfulness constraints specifically for features are instantiated in grammar, (e.g. MAX[F], DEP[F], UNIFORMITY[F], etc.). On the other hand, when features are dependents of segments, featural faithfulness is evaluated on the basis of the set of segmental faithfulness constraints, namely, IDENT[F]. I claim that features are independent elements of segments; therefore, an independent set of faithfulness constraints for features is established. The reason behind this decision is that some of the effect of OCP on features cannot be explained without the independent set of faithfulness constraints for features.

First, let us consider why we cannot deal with the typology when we treat features as dependents on segments by examining a hypothetical example /tot/. If features are treated as dependents, then we have only IDENT[F], thus featural faithfulness is regulated through segments. There are at least four output candidates for the input /tot/: (a) OCP violation; (b) Feature Fusion; (c) Feature Deletion and Insertion; and (d) Feature Deletion and Segmental Deletion. Among them, the candidate of featural fusion will always be optimal regardless of the ranking of the constraints, as seen in (29).

(29) a hypothetical tableau:

/t o t/ **| |** **[cor][cor]**	OCP[F1]	IDENT[F1]	IDENT[F2]	MAX-IO
a. OCP violation **[t o t]** **| |** **[cor][cor]**	*			
b. Feature Fusion[7] **[t o t]** **\ /** **[cor]**				
c. Feature Deletion and Feature Insertion **[lab]** **|** **[t o p]** **|** **[cor]**		*	*	
d. Feature Deletion and Segmental Deletion **[t o]** **|** **[cor]**				*

Since candidate (b) in which featural fusion takes place does not violate any constraint, it will always be optimal regardless of the ranking of constraints. If this were true, then there should be no language in which OCP violation, feature deletion & segmental deletion, or feature deletion & feature insertion is observed. Thus, we

7 I consider that the structure of featural fusion will not result in the OCP violation, because I consider that the two coronal features are fused into one. Thus, the structure does not contain [cor][cor] anymore.

cannot explain the typology of the OCP effects on features as long as features are treated as dependents on segments.

Consequently, I treat features as independent elements of segments in order to introduce a full set of featural faithfulness constraints. Next, let us review how features are treated in previous studies, and how featural faithfulness constraints are considered.

As mentioned in chapter 1, McCarthy (1996a, 1997a) indicates that there are two possible ways for us to treat features within the Optimality Theory: one is to treat features as dependent elements on the segments, and the other is to treat them as independent components. Following McCarthy, I call the former "features as attributes", and the latter "features as entities".

It is very important to determine if features should be treated as attributes or as entities when we consider the featural faithfulness constraints. If features are attributes, then, we could expect only the set of faithfulness constraints established for the segments introduced in section 2.3.2.1.

On the other hand, an independent full set of faithfulness constraints for the features could be instantiated, if features were entities. McCarthy (1996a, 1997a) indicates that correspondence relation holds only of segments if features are attributes, while it applies to segments and features if the latter are entities. Also, he warns that the independent set of featural faithfulness constraints should not be introduced unless additional constraints which can prohibit the features from moving freely around to other segments are also posited.

McCarthy's warning is based the tradition of the treatment of the features in autosegmental phonology. In the framework of autosegmental phonology, all autosegments are considered to hang on segments on different tiers. Dependency or independency of each autosegment, therefore, is determined by whether the autosegment can stand by itself without the existence of the segment on which it depends. In this sense, most of the suprasegments, and very few of the features such as [nasal] are regarded as independent elements because they are still retained after the sponsoring segment deletes. On the other hand, most of the features are considered to be dependent on segments because they disappear along with deletion of the sponsoring segment. That is why McCarthy claims that the treatment of the features as entities should not be freely introduced.

In spite of his claim, several researchers such as Itô, Mester, and Padgett (1995), Lamontagne and Rice (1995), Lombardi (1995a), McCarthy (1996a, 1997a), Causley (1997), and Morén and Miglio (1998) have indicated on the basis of various evidence that the features should be treated as entities so that faithfulness constraints for features can be instantiated. These researchers have shown that certain phenomena could not be explained unless features are treated as independent of segments.

Lombardi (1995a) and Lamontagne and Rice (1995) propose that both MAX[F] and DEP[F] are necessary to account for featural identity from the perspectives of both deletion and insertion of features.

(30) **MAX[F]** and **DEP[F]**

MAX[F]: Every feature in the input has a correspondent in the output (no featural deletion) (Lombardi 1995a, Lamontagne and Rice 1995)

DEP[F]: Every feature in the output has a correspondent in the input (no featural insertion) (Lombardi 1995a, Lamontagne and Rice 1995)

MAX[F] and DEP[F], not IDENT[F], constrain featural deletion and insertion, respectively. On the other hand, IDENT[F] requires that the corresponding segments share the same value for some feature. IDENT[F] is, therefore, violated whether it is due to deletion, due to insertion, or both , since in both cases the values for a feature are different between the corresponding segments.

McCarthy (1996a, 1997a) introduces two constraints which prohibit adding and deleting association lines. As McCarthy has pointed out, these two constraints specifically prevent the features from moving freely to other features.

(31) **No-Spread** and **No-Delink** (definitions from McCarthy 1997a:203)

No-Spread$_{S1\text{-}S2}$(τ, ς)

Let τ_i and ς_j stand for elements on distinct autosegmental tiers in two related phonological representations S_1 and S_2, where

τ_1 and $\varsigma_{1 \in}$ S_1,
τ_2 and $\varsigma_{2 \in}$ S_2,
τ_1 R τ_2, and
ς_1 R ς_2,

if τ_2 is associated with ς_2,
then τ_1 is associated with ς_1.
(Do not add association line)

No-Delink$_{S1\text{-}S2}$(τ, ς)

Let τ_i and ς_j stand for elements on distinct autosegmental tiers in two related phonological representations S_1 and S_2, where

τ_1 and $\varsigma_1 \in S_1$,
τ_2 and $\varsigma_2 \in S_2$,
τ_1 R τ_2, and
ς_1 R ς_2,

if τ_1 is associated with ς_1,
then τ_2 is associated with ς_2.
(Do not delete association line)

With these two constraints, a feature in the input cannot be spread to another segment, nor can it be delinked from the segment to which it belongs.

Itô, Mester, and Padgett (1995) illustrate featural deletion, insertion, and coalescence with constraints similar to those of Lombardi (1995a), Lamontagne and Rice (1995), and McCarthy (1996a, 1997a).

(32) Feature Faithfulness (Itô, Mester, and Padgett 1995: 586)

Parse Feat: All input features are parsed.

Fill Feat: All output features are part of the input.

Parse Link: All input association relations are kept.

Fill Link: All output association relations are part of the input.

Causley (1997) tries to generalize featural faithfulness constraints based on these previous studies. She argues that it is necessary for features to stand in a correspondence relation with one another; hence, the set of faithfulness constraints only for the segments are not sufficient to account for featural deletion, insertion, fusion, skipping, and so on. She claims that a full set of correspondence constraints for the segments are also multiplied for the features.

Table III: Correspondence Constraints for Features
(modified from Causley 1997:11)

Type of Constraint	prohibits
MAX[F]	Featural Deletion
DEP[F]	Featural Epenthesis
(I,O)-CONTIGUITY[F]	Featural Skipping (I-CONTIG[F]) Feature Intrusion (O-CONTIG[F])
ANCHOR[F]	Featural Misalignment
LINEARITY[F]	Featural Metathesis
UNIFORMITY[F]	Featural Coalescence
INTEGRITY[F]	Featural Splitting
No-Spread	Insertion of association line
No-Delink	Deletion of association line

Let us summarize what we have observed in the previous studies on features as entities. Since only deletion, insertion, and fusion of features will be discussed in the typology, I will concentrate on the constraints relative to these phenomena. As mentioned above, although McCarthy (1996a, 1997a) clearly indicates that features must not be completely independent of the segments, several studies such as Lombardi (1995a), McCarthy (1996a, 1997a), Causley (1997), Itô, Mester and Padgett (1995) show that features must be treated as entities.

Let us examine what kinds of faithfulness constraints are involved when we consider features as attributes. In Correspondence Theory, McCarthy and Prince (1995) propose a faithfulness constraint in which features are dependent on segments, i.e. IDENT[F]. This constraint is respected when the corresponding segments have the same values for the feature. When a corresponding segment deletes, then, the constraint is vacuously satisfied regardless of the absence or presence of the feature. Thus, featural deletion and featural insertion can be constrained iff the segments are present. In this sense, there is no constraint which can prohibit featural fusion.

Now, what constraints restrict the featural deletion, insertion, or fusion, when we treat the features as entities? Lombardi (1995a), and Lomontagne and Rice (1995) propose MAX[F] which prohibits featural deletion, and DEP[F] which forbids featural insertion. Both constraints can account for the featural deletion and insertion regardless of the presence of the corresponding segment.

McCarthy (1996a, 1997a) also uses MAX[F] and DEP[F]. In addition, he introduces NODELINK and NOSPREAD. These are the faithfulness constraints for association lines between segments and features. The former prohibits deleting association lines, and the latter prohibits adding them. I will actually introduce such faithfulness constraints for association lines between the segments to account for Japanese Rendaku in chapter 5.

These two constraints can also account for featural fusion, because in the case of featural fusion, it is not the feature which deletes or spreads, but the association line which changes. Itô, Mester, and Padgett's (1995) featural faithfulness constraints are very similar to McCarthy's.

Causley (1997) argues that the full set of faithfulness constraints must be instantiated. According to her, therefore, the segmental and the featural sets are completely comparable. Hence, in her approach, MAX[F], DEP[F], UNIFORMITY[F], etc. play the same roles as MAX, DEP, UNIFORMITY, etc. for the segments. For instance, segmental coalescence can be explained by violating UNIFORMITY, not

MAX. In the same way, in the case of featural fusion, MAX[F] is not violated, but UNIFORMITY [F] is.

Let us summarize what we have discussed so far with the following table. The following table shows which faithfulness constraints are violated in which featural alternations, and which researchers propose which constraints.

Table IV: Proposed Featural Faithfulness Constraints

	constraints when features are as attributes	constraints when features are as entities			
	McCarthy & Prince (1995)	Lombardi (1995a), Lamontagne & Rice (1995)	McCarthy (1996a, 1997a)	Itô, Mester, & Padgett (1995)	Causley (1997)
no featural deletion	IDENT[F]	MAX[F]	MAX[F]	Parse Feat	MAX[F]
no featural insertion	IDENT[F]	DEP[F]	DEP[F]	Fill Feat	DEP[F]
no featural fusion	N/A		NODELINK + NOSPREAD	Parse Link + Fill Link	UNIFORMITY [F]

Judging from the summary in Table IV, features must be treated as entities especially when featural deletion and insertion must be differentiated, and when featural fusion must be accounted for. If featural insertion, deletion, and fusion were not explained, then, the typology of the repair strategies for the OCP on features introduced in section 2.2.1 could not be accounted for, either.

On the basis of the discussion of the necessity of independent set of faithfulness constraints to account for the typology as mentioned above, and following previous research, I treat features as entities in this dissertation. The following section will discuss the constraint interaction that predicts each type of repair strategy.

2.3.3 Constraint Interaction to Predict the Typology

In the previous section, I have stated that I will treat features as entities so that a full set of correspondence constraints are instantiated for features. Now, I will investigate a constraint ranking to predict each type of language in the typology introduced in section 2.2.1.

There are slight differences among the previous studies on the kind of constraint(s) to restrict featural insertion, deletion, and fusion. In this thesis, I will use MAX[F] and DEP[F] for featural insertion and deletion, respectively, because they have been widely used. Also, I will adopt Causley's UNIFORMITY [F] to constrain featural fusion although McCarthy's (1996a, 1997a) NOSPREAD and NODELINK or Itô, Mester, and Padgett's (1995) Parse Link and Fill Link also seems to work to account for featural fusion. My choice is motivated by the devise to treat featural fusion in a parallel way to segmental coalescence. Segmental fusion will result in violating UNIFORMITY, while it will satisfy MAX-IO. Similarly, I consider that featural fusion will violate UNIFORMITY[F], but will satisfy MAX[F].

Let us consider featural fusion with an actual structure:

(33)

```
 / R1  R2 /   →   [ R1  R2 ]
    |α   |β            \α  /β
 [cor]i [cor]j        [cor]i,j
```

As illustrated in (33), when the two [cor] features are fused, none of the coronal features and none of the association lines delete. Both of the coronal features i and j in the input have the correspondents in the output; therefore, there is no MAX[cor] violation in this alternation. Also, both of the association lines α and β have the correspondents in the output; hence, there is no violation of the faithfulness constraints for association lines. Consequently, only the constraint violated in this structure is UNIFORMITY[F] which prohibits the output feature from having the multiple correspondents in the output.

Thus, I use UNIFORMITY[F] to constrain featural fusion since the structure of featural fusion is exactly parallel to the structure of segmental fusion: $/R_1\ R_2/ \rightarrow [R_{1,2}]$.

The full range of featural faithfulness constraints (MAX[F], DEP[F], UNIFORMITY[F] etc.) are crucial to analyze the effect of the OCP on features. Particularly, I will focus upon the interaction of OCP[F], MAX[F], DEP[F], UNIFORMITY[F], and MAX-IO to predict each type of language. Before going on to the actual analysis, the definition of each constraint is reviewed below.

(34) Constraints for Typology:

(a) OCP[F] = $[*FF]_{DOMAIN}$
Adjacent identical features are prohibited (Leben 1973; Goldsmith 1976; Mester 1986; McCarthy 1986).

(b) MAX[F]: Every feature of the input has a correspondent in the output (no featural deletion) (Lombardi 1995a, Lamontagne and Rice 1995).

(c) DEP[F]: Every feature of the output has a correspondent in the input (no featural insertion) (Lombardi 1995a, Lamontagne and Rice 1995).

(d) UNIFORMITY[F]: No feature of the output has multiple correspondents in the input (no featural fusion) (McCarthy and Prince 1995, Causley 1997).

(e) MAX-IO: Every segment of the input has a correspondent in the output.
(no segmental deletion) (McCarthy and Prince 1995).

2.3.3.1 Type 1 Language: OCP Violation

As discussed in section 2.2.1.1, OCP violation is observed in a Type 1 language. Let us establish which constraint ranking yields this type of language. Since the OCP violation is preferable to any other alternations here, we assume that all the faithfulness constraints must outrank the constraint for the OCP.

(35). Type 1

/t o t/ | | [cor][cor]	MAX [cor]	DEP[lab]	UNIFORM-ITY[cor]	MAX-IO	OCP[cor]
☞a. [t o t] | | [cor][cor] OCP violation					*
b. [t o t] \ / [cor] feature fusion			*!		
c. [lab] | [t o p] | [cor] feature deletion & insertion	*!	*!			
d. [t o] | [cor] feature & segment deletion	*!			*!	

The tableau in (35) shows that the given ranking predicts a language in which an OCP violation is observed. However, it is not crucial that all the faithfulness in tableau (35) outrank the OCP to yield a Type 1 language. A Type 1 language will also result from either the ranking, MAX[F], UNIFORMITY[F] >> OCP[F], or the ranking DEP[F], MAX-IO, UNIFORMITY[F] >> OCP[F] as the following two tableaux show.

(36) MAX[F], UNIFORMITY[F] >> OCP[F] for Type 1:

/t o t/ | | **[cor][cor]**	MAX[cor]	UNIFORMITY [cor]	OCP[cor]
☞a. **[t o t]** | | **[cor][cor]** OCP violation			
b. **[t o t]** \ / **[cor]** feature fusion		*!	
c. **[lab]** | **[t o p]** | **[cor]** feature deletion & insertion	*!		
d. **[t o]** | **[cor]** feature & segment deletion	*!		

(37) DEP[F], UNIFORMITY[F], MAX-IO >> OCP[F] for Type 1:

<table>
<tr><th>/t o t/
| |
[cor][cor]</th><th>DEP[lab]</th><th>UNIFORM-
ITY[cor]</th><th>MAX-IO</th><th>OCP
[cor]</th></tr>
<tr><td>☞a. [t o t]
| |
[cor][cor]
OCP violation</td><td></td><td></td><td></td><td>*</td></tr>
<tr><td>b. [t o t]
\ /
[cor]
feature fusion</td><td></td><td>*!</td><td></td><td></td></tr>
<tr><td>c. [lab]
|
[t o p]
|
[cor]
feature deletion
& insertion</td><td>*!</td><td></td><td></td><td></td></tr>
<tr><td>d. [t o]
|
[cor]
feature & segment
deletion</td><td></td><td></td><td>*!</td><td></td></tr>
</table>

Thus, a Type 1 language is result from at least either {MAX[F], UNIFORMITY[F]} or {DEP[F], MAX-IO, UNIFORMITY[F]} outranking OCP[F]. Therefore, the following rankings are established for Type 1.

(38). Constraint Ranking for Type 1:

(a)

MAX[F], UNIFORMITY[F]
|________|
|
OCP[F]

(b)

DEP[F], UNIFORMITY[F], MAX-IO
|________|______________|
|
OCP[F]

2.3.3.2 Type 2 Language: Feature Fusion

In this type of language, the two features are fused so as not to violate the OCP constraint on features. Thus, we assume that only UNIFORMITY[F] is lower ranked than the OCP constraint and the other faithfulness constraints in this type of language.

(39) Type 2

/t o t/ | | **[cor][cor]**	MAX [cor]	DEP [lab]	OCP [cor]	MAX-IO	UNIFORM-ITY[cor]
a. **[t o t]** | | **[cor][cor]** OCP violation			*!		
☞b. **[t o t]** \ / **[cor]** feature fusion					*
c. **[lab]** | **[t o p]** | **[cor]** feature deletion & insertion	*!	*!			
d. **[t o]** | **[cor]** feature & segment deletion	*!			*!	

In candidate (39b), MAX[cor] is not violated, because both input coronal features have some correspondents in the output. Thus, it violates only UNIFORMITY[F].

The ranking given in (39) predicts a Type 2 language. However, not all the faithfulness constraints in this tableau must be higher ranked than UNIFORMITY[F]. A Type 2 language is predicted by either the ranking MAX[cor], OCP[cor], MAX-IO >> UNIFORMITY[F] or the ranking DEP[lab], OCP[cor] >> UNIFORMITY[F] as the following two tableaux show.

(40) DEP[lab], OCP[F], MAX-IO >> UNIFORMITY[F] for Type 2:

/t o t/ | | [cor][cor]	DEP[lab]	OCP[cor]	MAX-IO	UNIFORM-ITY[cor]
a. [t o t] | | [cor][cor] OCP violation		*!		
☞ b. [t o t] \ / [cor] feature fusion				*
c. [lab] | [t o p] | [cor] feature deletion & insertion	*!			
d. [t o] | [cor] feature & segment deletion			*!	

(41) MAX[F], OCP[F] >> UNIFORMITY[F] for Type 2:

/t o t/ | | **[cor][cor]**	MAX[cor]	OCP [cor]	UNIFORMITY [cor]
a. **[t o t]** | | **[cor][cor]** OCP violation		*!	
☞ b. **[t o t]** \ / **[cor]** feature fusion			*
c. **[lab]** | **[t o p]** | **[cor]** feature deletion & insertion	*!		
d. **[t o]** | **[cor]** feature & segment deletion	*!		

Thus, the ranking in (40) or in (41) accounts for a Type 2 language.

(42) Constraint Ranking for Type 2:

(a)

```
DEP[F],    OCP[F],      MAX-IO
   |__________|__________|
              |
       UNIFORMITY[F]
```

(b)

```
MAX[F],    OCP[F]
  |__________|
        |
 UNIFORMITY[F]
```

2.3.3.3 Type 3 Language: Feature Deletion and Insertion

In this type of language, one of the two adjacent features is simply deleted to avoid the OCP violation. Moreover, a new feature is inserted to avoid an ill-formed structure due to the feature deletion. This is illustrated in (43).

(43) Type 3

/t o t/ | | **[cor][cor]**	OCP [cor]	MAX-IO	UNIFORMITY[cor]	MAX [cor]	DEP[lab]
a. **[t o t]** | | **[cor][cor]** OCP violation	*!				
b. **[t o t]** \ / **[cor]** feature fusion			*!		
actual winner c. **[lab]** | **[t o p]** | **[cor]** feature deletion & insertion				*	*!
d. **[t o]** | **[cor]** feature & segment deletion		*!		*	
*☞ e. **[t o t]** | **[cor]** only deletion of feature				*	

In (43), based on the ranking established thus far, candidate (e) in which only the coronal feature deletes wrongly wins. To obtain the correct output, an additional constraint: HAVEPLACE must be introduced and ranked above MAX[cor].

Padgett (1994, 1995a) indicates that a segment without ***Place*** or ***Manner*** is ill-formed; therefore, he proposes a constraint against a placeless segment, **HAVEPLACE**. Let us reanalyze the tableau (43) with the constraint HAVEPLACE.

(44) TYPE 3 with HAVEPLACE:

/t o t/ | | **[cor][cor]**	OCP [cor]	MAX-IO	UNIFORM-ITY [cor]	HAVE PLACE	MAX [cor]	DEP [lab]
a. **[t o t]** | | **[cor][cor]** OCP violation	*!					
b. **[t o t]** \ / **[cor]** feature fusion			*!			
☞ c. **[lab]** | **[t o p]** | **[cor]** feature deletion & insertion					*	*
d. **[t o]** | **[cor]** feature & segment deletion		*!			*	
e. **[t o t]** | **[cor]** only deletion of feature				*!	*	

In tableau (44), the crucial ranking, HAVEPLACE >> DEP[cor] brings forth the correct output.

As the analysis for type 3 above shows, an additional constraint such as HAVEPLACE is necessary. Hence, the following ranking is established for Type 3.

(45) Constraint Ranking for Type 3:

```
OCP[F],UNIFORMITY[F], MAX-IO   HAVEPLACE
     |_________|____________|       |
            _____|_________          |
            |              |         |
         MAX[F]           DEP[F]
```

2.3.3.4 Type 4 Language: Featural and Segmental Deletion

In this type, the ill-formedness caused by featural deletion is repaired by deleting the entire segment. Since we are using MAX[F], not IDENT[F], deletion of the segment will not result in satisfying the constraint for featural deletion. Both segmental and featural faithfulness constraints are violated.

(46) Type 4

/t o t/ | | **[cor][cor]**	OCP [cor]	DEP [lab]	UNIFORM-ITY [cor]	MAX [cor]	MAX-IO
a. **[t o t]** | | **[cor][cor]** OCP violation	*!				
b. **[t o t]** \ / **[cor]** feature fusion			*!		
c. **[lab]** | **[t o p]** | **[cor]** feature deletion & insertion		*!		*	
d. actual winner **[t o]** | **[cor]** feature & segment deletion				*	* !
*☞ e. **[t o t]** | **[cor]** only deletion of feature				*	

In (46), again, candidate (e) in which only feature deletion takes place incorrectly becomes optimal. Therefore, HAVEPLACE also has to be higher ranked than MAX-IO in this type of language. Also, it is not the violation of MAX[F], but rather of DEP[F], which penalizes candidate (c) in Type 4.

Let us see the analysis with the additional constraint, HAVEPLACE.

(47) TYPE 4 with HAVEPLACE:

/t o t/ | | [cor][cor]	OCP [cor]	DEP [lab]	UNIFORM-ITY [cor]	HAVE PLACE	MAX [cor]	MAX-IO
a. [t o t] | | [cor][cor] OCP violation	*!					
b. [t o t] \ / [cor] feature fusion			*!			
c. [lab] | [t o p] | [cor] feature deletion & insertion		*!			*	
☞ d. [t o] | [cor] feature & segment deletion					*	*
e. [t o t] | [cor] only deletion of feature				*!	*	

Thus, in tableau (47), the crucial ranking HAVEPLACE >> MAX-IO correctly accounts for the actual winner.

(48) Constraint Ranking for Type 4:

```
OCP[F], UNIFORMITY[F], DEP[F]         HAVEPLACE
|__________|______________|                |
          ______|__________________        |
          |                       |        |
       MAX[F]                    MAX-IO
```

Thus, all four types of languages are predicted on the basis of the constraint interactions of OCP[F], UNIFORMITY[F], DEP[F], MAX[F], and MAX-IO.

In addition, I have introduced an additional structural constraint HAVEPLACE needed for the correct analysis. Although the four constraint rankings given above are the basic rankings for each type of language in the typology, we have to take the interaction of other constraints into consideration to elucidate actual language data.

2.4 Summary

In this chapter, I discussed how we account for the typology of OCP effects on features within the framework of Optimality Theory. In section 2.1, I first reviewed the previous research on OCP both in the framework of autosegmental phonology and of Optimality Theory, and considered the relevant constraints for the effects of the OCP. Following the self-conjunction approach to OCP, I made clear my position that OCP is derived from self-conjoined markedness constraints.

In section 2.2, I investigated the classification of languages with respect to the effects of the OCP on features. Although my typological categorization of languages were built on Yip's (1988) proposal, I reorganized her classification in order to classify languages based on the OCP effects specifically on features. I eliminated epenthesis from Yip's claim, and reconsidered dissimilation, assimilation, and deletion.

In section 2.3, I discussed the kind of constraints interacting in each type of language in the typology established in section 2.2. I especially focused on the interaction among the featural faithfulness constraints, the OCP and the markedness constraints. Through the discussion of the interaction of those constraints, I showed that features must be treated as independent elements of segments so that a full set of faithfulness constraints specific to features are instantiated in grammar. I showed that all the introduced constraints in the discussion are needed to account for the typology of the OCP effects on features, and none of the constraints are arbitrary. I clearly demonstrated that the different rankings of those constraints represent the different types of languages.

Thus, I delineated the classification of languages regarding the effect of the OCP on features, and examined the tools, i.e. the constraints, crucial to account for each type of the language. With the constraints examined in this chapter, the next chapter will analyze the data of the actual languages. One from each type will be examined in order to confirm the validity of the predicted rankings discussed in this chapter.