

CHAPTER 1

CHAIN SHIFTS AS CONTRAST TRANSFORMATION

1.1 Statement of the Problem

This dissertation provides an account of chain shift mappings in terms of preserving contrasts (cf. Kirchner 1996, Kisseberth 1976). In a phonological chain shift underlying /A/ maps onto surface [B] and underlying /B/ maps onto surface [C]. Thus, there is a chain shift effect of the form $A \rightarrow B \rightarrow C$. Finnish vowel shift (Anttila 1995, Harrikari 1999, 2000, Lehtinen 1967, McCawley 1964) provides an example.

In Finnish, before the plural suffix *-i* (similarly before the past tense marker *-i*), long low vowels shorten ($/aa/ \rightarrow [a]$), short low vowels undergo rounding (and raising) ($/a/ \rightarrow [o]$), and short round vowels surface unchanged ($/o/ \rightarrow [o]$). Thus, we have the following chain shift effect:

(1) Finnish chain shift

$aa \rightarrow a \rightarrow o$

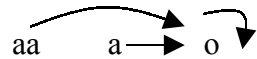
Some examples are given in (2).

(2)		<i>singular nominative</i>		<i>plural essive</i>
	$/aa/ \rightarrow [a]$	kattaa maa	‘roof’ ‘earth’	katta-i-na ma-i-na
	$/a/ \rightarrow [o]$	kissa vapa	‘cat’ ‘fishing rod’	kisso-i-na vapo-i-na
	$/o/ \rightarrow [o]$	talo pelko	‘house’ ‘fear’	talo-i-na pelko-i-na

The key issue is that in Finnish forms with underlying long low vowels shorten but do not round (/aa/→[a],*[o]), but forms with underlying short low vowels undergo rounding in the same context (/a/→[o]).

Chain shifts present a challenge to Optimality Theory in its original form. OT is *output-oriented*: phonological processes like /aai/→[ai] and /ai/→[oi] bring output forms into conformity with high-ranking markedness constraints. The existence of these processes in Finnish indicates that two markedness constraints, informally *aai and *ai, dominate antagonistic faithfulness constraints. But with both markedness constraints high-ranked, we expect underlying /aai/ to go all the way to [oi], thereby satisfying both markedness constraints. The expected but incorrect mapping is illustrated in (3).

(3) Expected mapping (cf. (1))



Even though chain shifts are problematic for classic Optimality Theory, they are part of a synchronic mechanism of the grammar and we need to be able to account for them formally. Some examples of vowel height chain shifts come from Bassá (Bantu; Schmidt 1996), Gbanu (Niger-Congo; Bradshaw 1996), Kikuria (Bantu; Chacha and Odden 1994), Lena Spanish (Hualde 1989), Nzɛbi (Clements 1991), Servigliano Italian (Kaze 1991). These are mostly raising mappings (Parkinson 1996). Some examples of consonantal chain shifts come from Southern Paiute (Sapir 1930, McLaughlin 1984), Toba Batak (Hayes 1986), Estonian (Utan 1970), Finnish (Utan 1970), Irish (Ní Chiosá in 1991). These are mostly lenition mappings on either voicing or consonantal stricture scale (Gnanadesikan 1997). See (4) and (5), respectively.

- (4) Vowel shifts (Clements 1991, Labov 1994)
 a. New Zealand (Labov 1994): $\text{æ} \rightarrow \text{e} \rightarrow \text{i} \rightarrow \text{i}^{\text{h}}$
 b. Nzɛbi (Bantu: Clements 1991): $\text{a} \rightarrow \text{ɛ} \rightarrow \text{e} \rightarrow \text{i}, \text{ɔ} \rightarrow \text{o} \rightarrow \text{u}$
- (5) Consonantal shifts (Ulan 1970)
 a. Southern Paiute (Uto-Aztecan, Sapir 1930): $\text{pp} \rightarrow \text{p} \rightarrow \text{v}$
 b. Toba Batak (Austronesian, Hayes 1986): $\text{np} \rightarrow \text{pp} \rightarrow \text{ʔp}$

According to Gnanadesikan (1997) and Kirchner (1996), the solution to chain shift mappings in OT lies in an enriched theory of faithfulness. Both researchers propose special types of faithfulness constraints that block two-step movements like /aai/→[oi], thereby accounting for the discrepancy in phonological mappings between identical derived and underlying segments. Kirchner uses locally-conjoined faithfulness constraints, whereas Gnanadesikan distinguishes between classical IDENT-type constraints and novel IDENT-ADJACENT-type constraints on some scale of similarity.¹

In this dissertation, I will explore an alternative explanation for chain shifts that has implications going well beyond this phenomenon. The explanation starts from the observation that chain shifts always preserve one underlying contrast at the expense of neutralizing another underlying contrast. In Finnish, the contrast between underlying /aai/ and /ai/, originally one of length, is preserved, albeit in a different form – as a rounding contrast (underlying /aai/ vs. /ai/ surface [ai] vs. [oi]). The contrast between underlying /ai/ and /oi/, the original rounding contrast, is lost (both become [oi]). Thus:

- | | | | |
|-----|-------------------|---|-------------------|
| (6) | <u>Input</u> | | <u>Output</u> |
| | length contrast | → | rounding contrast |
| | rounding contrast | → | neutralized |

¹ For a discussion of Gnanadesikan's and Kirchner's proposals see chapter 4.

Preservation of one contrast taking precedence over preservation of another contrast will be referred to as *contrast transformation*.

There are numerous examples of contrast transformation across languages. Some are given in (7). These are various ways of preserving the obstruent voicing contrast that cannot be realized as such. The examples are from Polish, Friulian, spoken in northern Italy, and dialects of American English.

- (7) Preservation of the obstruent voicing contrast
- a. Interaction of final devoicing and vowel raising in Polish (Gussmann 1980)
/ko**z**/ vs. /ko**s**/ → [ku**s**] vs. [k**o**s] *by vowel height*
'goats' vs. 'scythes' (gen.)
 - b. Interaction of final devoicing and vowel lengthening in Friulian (Repetti 1994)
/la**d**/ vs. /la**t**/ → [la**:t**] vs. [la**t**] *by vowel length*
'gone' (m.) vs. 'milk'
 - c. Interaction of intervocalic voicing and vowel lengthening in Am. English
/ra**y**de**r**/ vs. /ra**y**te**r**/ → [ra**:y**De**r**] vs. [ra**y**De**r**] *by vowel length*
'rider' vs. 'writer'

In Polish, see (7a), the underlying obstruent voicing contrast in word-final position is preserved on the surface despite final devoicing and manifested as vowel height contrast. The vowel is high before underlyingly voiced obstruents and mid before their voiceless counterparts. In Friulian, see (7b), the obstruent voicing contrast is manifested as surface contrast in vowel length. In American English (7c), instead of final devoicing there is a process of intervocalic flapping. The obstruent voicing contrast is preserved despite intervocalic flapping and represented by vowel length.

Contrast transformation is common cross-linguistically. The main question in this work is where contrast preservation fits into the grammar, whether it follows from other components of the grammar or exists as a primitive. In generative phonology, contrast

preservation is an epiphenomenon of rule application. Whether contrasts are preserved or neutralized follows from what rules there are and how they apply in a given language. In standard Optimality Theory (OT), contrast preservation is also a derivative. It follows from the interaction of markedness and faithfulness constraints that do not themselves refer to contrast. The proposal here is different from both generative phonology and standard Optimality Theory. It is proposed that contrast preservation exists as an independent principle in the grammar, which in the framework of Optimality Theory is formulated as a family of rankable and violable constraints on preserving contrasts. I will refer to this proposal as *Contrast Preservation Theory* (PC theory).

The next section (section 1.2) presents the elements of the proposal. Section 1.3 illustrates the proposal with a simple case of neutralization and the lack of it. Subsequent sections (sections 1.4-1.5) show how the proposal can be applied to analyze Finnish and other chain shifts.

1.2 PC Theory

To account for phonological mappings that involve contrast transformation like chain shifts, I will propose a modification of Optimality Theory (Prince and Smolensky 1993), called *Contrast Preservation Theory* (PC theory). In PC theory contrast preservation is not just a phenomenon but a formal property of the grammar (cf. Flemming 1995, 1996, Padgett 1997, 2000). It is formalized as competing constraints on preserving contrasts.

PC theory also contributes to our understanding of opacity (Kiparsky 1973, 1985, 1993; Kisseberth 1976; McCarthy 1999; Rubach 1984). Opaque processes, of which chain shifts are an example, are problematic for standard OT. In standard OT a process

applies when markedness outranks conflicting faithfulness and is blocked with the opposite ranking. This is true for transparent processes. But opaque alternations do not comply with those rankings. Opaque processes apply even though markedness is low-ranked and are blocked despite high-ranked markedness. Standard OT finds those problematic. PC provides an explanation.

A central finding of PC theory is that opaque phonological alternations trade one distinctive opposition for another, preserving contrast but in a different way than in the underlying form. Based on this observation, PC theory proposes that opaque processes can be explained by a high-ranking requirement on preserving contrast.

PC makes a novel prediction as to what can force or block a phonological process. In PC phonological mappings are evaluated together, and thus one mapping can force or block another mapping in the same system for reasons of contrast. This prediction is different from other approaches to phonology, where mappings are evaluated in isolation.² In the previous approaches, mappings cannot directly activate or block one another. I will argue that the predictions of PC theory are superior to previous approaches and I will illustrate them using the example of the Finnish chain shift.

By formulating contrast preservation as an imperative in a phonological system, PC theory provides an explanation for opaque processes (e.g., chain shifts) and explains transparent and opaque phonological processes in a uniform manner with no additional

² In previous approaches, there are ways to encode morphological relatedness between forms such that forms can influence one another. In standard OT, relatedness between forms is encoded in the form of faithfulness constraints on Output-Output correspondence (Benua 1997, Burzio 1998), Base-Reduplicant Identity (McCarthy & Prince 1995), Paradigm Uniformity (Kenstowicz 1996, Steriade 2000) or morphological Anti-faithfulness (Alderete 1999, Horwood 2001). In derivational OT, relatedness between forms is encoded in underlying representations and by means of rule ordering. Yet in none of the approaches is mapping interaction stated explicitly, as it is in PC theory.

mechanisms required. This is different from previous OT approaches to chain shifts (see chapter 4).

The next section (1.2.1) shows how to form a candidate in PC theory, and the following section (1.2.2) describes the constraints.

1.2.1 A Scenario

The central claim of PC theory is that there exist anti-neutralization PRESERVE CONTRAST constraints. Constraints on contrast preservation can only be formalized under the assumption that no /input/→[output] mapping takes place in isolation; all such mappings are part of a system (cf. Flemming 1995, Padgett 1997). Formally, in the OT framework this must mean that candidates are sets of mappings, which I will call *scenarios*. The main idea is that mappings influence one another. A mapping can block or force another mapping in a system. The claim is that chain shifts can be understood as part of a system of mappings (a scenario).

1.2.1.1 The Logic

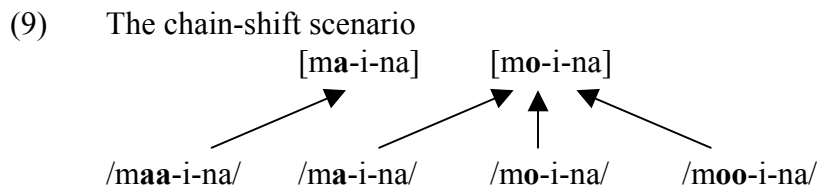
A scenario represents interaction between mappings. Let us first look at a scenario in Finnish. In Finnish, the logic behind a scenario is as follows. Underlying /aa/ undergoes shortening before -i and maps onto [a]. This forces underlying /a/ in the same context to move away - it undergoes rounding and maps onto [o]. By rounding, the low vowels distinct in length, /aa/ and /a/, do not neutralize on the surface. They map onto distinct outputs. This has consequences for the system of mappings. The length merger is avoided, but there is another merger that takes place as a consequence. Due to rounding, there is a merger between underlying /a/ and /o/. They both map onto [o].³

³ It is important that the mappings all take place in the same context.

There is another mapping in Finnish that also maps onto [o] and thus is part of the interacting mappings. This is the mapping for underlying /oo/. In Finnish, underlying /oo/ before /i/ shortens with no change in height (/oo/→[o]) and thus merges with the short vowels /a/ and /o/. Some examples of the mapping for underlying /oo/ are given below.

- (8) Shortening of /oo/
- | | | |
|--------|-----------|------------|
| ehtoo | ‘evening’ | ehto-i-na |
| tienuo | ‘area’ | tienu-i-na |

Altogether, the four mappings constitute part of the chain-shift scenario in Finnish: /aai/→[ai], /ai/→[oi], /oi/→[oi], and /ooi/→[oi]. This is shown below.



The actual chain-shift scenario competes with other scenarios in the same candidate set. In PC theory, each scenario is a candidate, and rankable constraints determine which scenario is optimal. Some examples of scenarios in a candidate set are given below. Scenarios represent languages with various mapping coexistence patterns.

- (10) Some scenarios in a candidate set

A. Identity scenario (Identity map)	B. Transparent scenario (Shortening)	C. Chain-shift scenario (Shortening & rounding)
[aai] [ai] [ooi] [oi] ↑ ↑ ↑ ↑ /aai/ /ai/ /ooi/ /oi/	[ai] [oi] ↗ ↑ ↗ ↑ /aai/ /ai/ /ooi/ /oi/	[ai] [oi] ↘ ↗ ↗ ↑ /aai/ /ai/ /ooi/ /oi/

In the identity scenario, each input maps onto an identical output. In the transparent

scenario, there is shortening but no rounding. In the actual scenario, there is both shortening and rounding, but rounding targets only underlying short vowels. (In (10), I show interacting processes only. As will be explained below, each scenario contains a whole space of mappings.)

1.2.1.2 Formalization

In rest of this section I describe how to formally determine the set of inputs and outputs of a scenario.

The Input. “Scenario-inputs” are returned by the function (the operator) *Gen* (cf. Prince and Smolensky 1993). The construction of scenario inputs is analogous to the role of *Gen* in correspondence theory (McCarthy & Prince 1995). Formally, for a given underlying form *Gen* returns “scenario-inputs.” Those are the inputs of each scenario in a candidate set. This is defined below.

(11) The role of *Gen* in PC
 $Gen(\text{underlying form}_i) \rightarrow \text{scenario-inputs}_i$

Gen takes an underlying form as its argument and returns a set of inputs as its value. The set of inputs returned by *Gen* (scenario-inputs) contains forms (string of segments) that can potentially interact. For an underlying form of length *n*, scenario-inputs contain all strings of length $0 \dots 2n+1$. I will now discuss the function *Gen* in more detail.

The input to *Gen* is a string of segments (a word) and the outputs are strings that are different from the input in P (phonological) properties. The P properties are distinctive phonological properties, such as *voicing*, *place*, *manner*, *length* etc. Let us put aside length differences for the moment and consider distinctive features other than segmental deletion and insertion. With that in mind, the set of scenario-inputs returned by

Gen for a given underlying form contains strings that consist of any sounds (bundles of P properties) and sound combinations that are logically possible.

- (12) The role of Gen
Gen (underlying form_i) → $\forall P$ (phonological) properties, \forall linear combinations of P,
 $\exists y$ such that $y \in \text{scenario-inputs}_i$

For example, the set of scenario-inputs for a three-segment underlying form *bad* contains *bad, bat, pat, ugh, k̡o* etc. This is illustrated below:

- (13) Inputs by feature changes
Gen (bad) → {bad, bat, pat, ugh, k̡o etc.}

In addition to featural differences, the input set also contains forms that differ from the underlying form in the number of segments. Those are forms that contain fewer segments than the underlying form (by deletion), including a null set, and forms that contain more segments (by epenthesis). Some examples for *bad* are *ba, a, bada* etc.

- (14) Inputs by deletion and insertion
Gen (bad) → {ba, a, bada etc.}

The way in which scenarios are evaluated, as will be discussed in the next section, demands that scenarios be finite. Therefore, epenthesis needs to be limited to only a certain number of segments. If epenthesis were not bounded, the input set would be infinite since we can add segments ad infinitum. To prevent unbounded insertion of segments, I must assume that there is a limit on the number of segments that can be added to the underlying form. There can be only as many segments added to the underlying form as there are segments in it plus 1. Formally, epenthesis takes place such that there is only one spot adjacent to each segment in a string of segments available for an epenthetic filler.

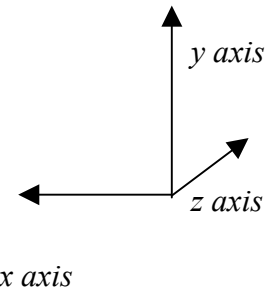
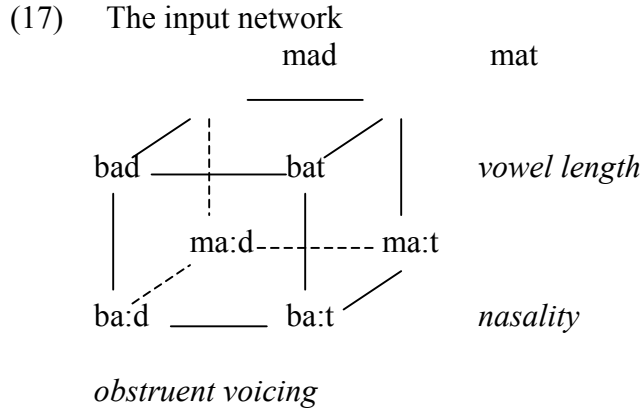
- (15) No unbounded epenthesis
Gen (bad) \rightarrow $_b_a_d_$, where $_$ represents possible sites of epenthesis.

In addition to forms with featural changes (see (13)) and deletion or insertion of segments (see (14)), there are forms where Gen changes more than one P property in one and the same form. For example, it combines a change in a P property, such as place or manner, with deletion or insertion of segments. Thus, altogether, the set of scenario-inputs returned by Gen for a three-segment underlying form *bad* contains, among other, forms *ba*, *a*, *ugh*, *ug* etc.

- (16) Altogether
Gen (bad) \rightarrow {*ba*, *a*, *bada*, *ugh*, *ug*, *pata* etc}

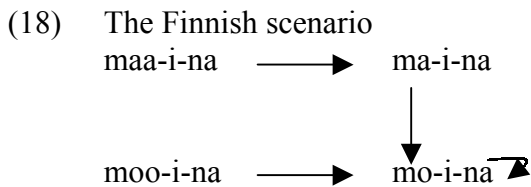
As a final comment on Gen, I would like to point out that Gen is a universal function, the same for all underlying forms, but scenario-inputs generated by Gen are not universal. The same scenario-inputs are generated for all underlying forms of a given length, length n , but different for underlying forms of different length. Given an underlying form of length, n , scenario-inputs generated by Gen are any strings $0\dots 2n+1$. The idea is that Gen generates all inputs that could possibly interact. Thus, the same scenario-inputs will be generated for *bad* as for *pat* or *ugh*. This set is different from the one for *pata*, a four-segment underlying form, though there is an overlap between the two scenario-inputs due to deletion and insertion of segments.

It is helpful to think of inputs as distributed in a multi-dimensional space. Inputs generated by Gen form a network, the dimensions of which are determined by the P properties. Here is a subset of the input network defined by three distinct P properties: (i) obstruent voicing in word-final position (x axis), (ii) vowel length (y axis), and (iii) nasality in word-initial position (z axis).



The various P properties define the space of inputs in a scenario. Since a scenario contains a set of input-output mappings, it is also necessary to define what outputs are included in the scenario.

The Output. I propose that a scenario is a mapping of the input set into itself. Thus, outputs in a scenario are a subset (possibly improper) of the input. There is nothing in the output of a scenario that is not also in the input. This limits the space of mappings that is evaluated. We do not go outside the set of forms that constitute scenario-inputs. Consider part of the Finnish scenario that consists of inputs minimally distinct in vowel length and vowel rounding. In the diagram below, arrows indicate input-output mappings. As shown here, outputs are drawn from the set of inputs.



Scenarios represent alternatives that compete for the status of the optimal scenario. Let us look at what scenarios compete with one another (what scenarios there are in a candidate set).

Scenarios as Candidates. Scenarios in a candidate set represent all mappings of the input set into itself. Thus, scenarios in a candidate set have the same inputs but differ in the set of outputs and/or input-output relations even if outputs are the same. As shown in (10), the identity scenario has a different set of outputs from the transparent and actual scenarios. The transparent and actual scenarios have the same outputs but differ on the input-output relations. Thus, there are two ways in which scenarios in a candidate set differ from each other: the set of outputs and/or input-output mappings.

The Size. Since each scenario represents one of the ways of mapping all the inputs contrasting in some feature(s) onto some output subset, the size of a scenario is determined by the input. Scenarios in a candidate set are of the same size. They contain the same number of inputs and thus the same number of input-output mappings. Since there is a finite number of oppositions and there is a limit on the length of words in a scenario, a scenario is finite. This is a crucial point, if contrast is to be evaluated.

A final note on scenario-inputs: The input set, as defined above, is finite since forms contrast with one another on some finite number of dimensions and insertion is bounded, but it is a big set. However, not all dimensions of contrast are equally informative. Some of the contrasts are always preserved. Similarly, some input pairs represent contrasts that are always lost (the latter type are inputs supplied by Richness of the Base). There are also contrasts that are preserved in some position but lost elsewhere.

In a tableau, not all of the relevant inputs will be shown. The inputs that will be shown in a tableau will consist of minimally distinct words (words that are distinct on a single P property in one and the same location), i.e., *bad* vs. *bat* are minimally distinct but not *bad* vs. *sat*. However, *bad* vs. *sat* will be included into the scenario in a tableau if

their minimally distinct counterparts, *bat* and *sad*, are also included. That is, for any form included in a tableau, there must be a form that is minimally contrastive on some property. Contrasts cross-classify. Given the way the set of inputs is generated, not all of those forms will be actual words of the language. This is what distinguishes my proposal from accounts where contrast preservation is strictly limited to avoidance of homophony, such as Crosswhite (1997), Steriade (1996). It is similar to Flemming (1995).⁴

This raises the question of how the process of learning takes place since the forms included into the scenario are possible words but not necessarily existing words of the language. The current hypothesis is that in the process of learning, the learner uses the actual words to establish a language particular constraint ranking and only then generalizes them to hypothetical forms (see *Predictions of the Theory*, chapter 4 for discussion).

The optimal scenario is chosen by the interaction among rankable and violable constraints. These are presented in the next section.

1.2.2 The Constraints

As defined in the previous section, a scenario is a space of mappings, with its dimensionality given by contrasting features. Outputs in the scenario are a subset of its inputs. Thus mappings are from the set of inputs onto its subset. (Mappings are output \subseteq input.) Scenarios in a candidate set compete for the status of the optimum. There are three aspects of scenario evaluation: contrast preservation (1.2.2.1), output well-formedness (1.2.2.2), and the degree of input-output disparity in a scenario (1.2.2.3).

⁴ The proposals of Padgett and Flemming will be discussed in *Predictions of the Theory*, chapter 4.

Contrast preservation compares scenarios for what types of contrasts are preserved or neutralized in surface forms, and at what cost. There are three aspects of contrast preservation that are evaluated: (i) the number of inputs involved in neutralizations, (ii) the number of outputs that are ambiguous as a result of neutralization, and (iii) the correspondence between input contrasts and output contrasts. Different aspects of contrast preservation take precedence in different languages, and thus different scenarios are optimal in those languages. (Section 1.2.2.1.)

In addition to contrast preservation, scenarios are compared for output well-formedness. Different scenarios may contain different outputs. In addition, since mappings are evaluated together, the same outputs may correspond to a different number of inputs in different scenarios. That is, scenarios differ not only in the types of outputs but also in the number of particular output forms. Both aspects of output well-formedness are evaluated. (Section 1.2.2.2.)

Finally, different scenarios may fare the same on contrast preservation and output well-formedness but they may differ in the degree of input-output disparity. The same interplay of contrasts and the same output forms can be achieved at various cost. The scenario with the smallest degree of disparity wins. Disparity is evaluated separately for different types of outputs in a scenario. (Section 1.2.2.3.)

The natural set of conditions on mappings is summarized below. Recall that these are conditions on contrast preservation, output well-formedness and input-output disparity, respectively. The condition on contrast preservation has three aspects to it, as mentioned above.

- (19) Conditions on mappings
 - i. Contrasts are preserved.
 - a. Inputs do not merge.
 - b. Outputs are not ambiguous.
 - c. Output contrasts correspond to identical input contrasts.
 - ii. Outputs are well-formed.
 - iii. Outputs and corresponding inputs are expressed in the same way.

As will become clear from the following discussion (see chapter 2 in particular), each of these conditions is indispensable for an effective comparison between scenarios in a candidate set.

In PC theory, each condition is formalized as a family of violable and rankable constraints. These are Preserve Contrast (PC constraints), tokenized markedness and generalized faithfulness. The latter is such that it does not distinguish between different types of identity. The following is a summary of the constraints.

- (20) Constraints in PC (cf (19))
 - i. Preserve Contrast (PC)
 - a. Input-oriented PC
 - b. Output-oriented PC
 - c. Relational PC
 - ii. Tokenized Markedness
 - iii. Generalized Faithfulness

Constraints in PC theory belong to two stages of Eval(uation). *Eval* is the evaluator function H that consists of the language-particular constraint hierarchy. In standard OT, Eval is a one-stage constraint ranking. In PC, it is proposed that there are two stages of Eval. PC and markedness constraints belong to stage 1 (H-eval₁), and generalized faithfulness to stage 2 (H-eval₂). This is to avoid redundancy between PC and generalized faithfulness. For example, both PC and generalized faithfulness can block a phonological process. Since Eval is subdivided into two stages, this means that generalized faithfulness constraints can apply only after PC and markedness get a chance

to apply. As a result, generalized faithfulness constraints deal with differences between scenarios that have not been determined in H-eval₁ by PC or markedness. The two stages of Eval are illustrated below.

- (21) Structure of PC grammar
- a. Gen (In_k) → {Scen₁, Scen₂, ... Scen_n}
 - b. H-eval₂ (H-eval₁ (Scen_i, 1 ≤ i < ∞)) → Scen_{real}

Where:
 H-eval₁ = PC and Markedness
 H-eval₂ = Generalized Faithfulness

Scenarios are first evaluated by PC and markedness in H-eval₁. The output of this evaluation process becomes an argument of H-eval₂ that consists of a language particular ranking of generalized faithfulness constraints.

In the following sections I discuss each constraint family in turn. I start out with PC constraints, followed by tokenized markedness, and generalized faithfulness. I then show how the constraints can be used to analyze Finnish and other chain shifts.

1.2.2.1 PC Constraints

PC constraints evaluate contrast. There exist three families of PRESERVECONTRAST constraints: input-oriented PC, output-oriented PC, and relational PC. Those constraints evaluate scenarios for whether and how they preserve underlying contrasts in surface forms. We need all of them because there are distinct forms of complexity that can inhere in different sets of mappings. The following sections describe the PC constraints in more detail. When introducing the constraints, I always compare two scenarios, one of which is the actual scenario from Finnish, while the other is a competing scenario from the same candidate set. Let us start out with input-oriented PC.

1.2.2.1.1 Input-oriented PC

Input-oriented PRESERVECONTRAST constraints, $PC_{IN}(P)$ for short, demand that pairs of words that contrast underlyingly in a given phonological property P contrast on the surface (not necessarily in P). Such constraints are defined in (22).

- (22) $PC_{IN}(P)$
For each pair of inputs contrasting in P that map onto the same output in a scenario, assign a violation mark. Formally, assign one mark for every pair of inputs, in_a and in_b , if in_a has P and in_b lacks P , $in_a \rightarrow out_k$, and $in_b \rightarrow out_k$.
“If inputs are distinct in P , they need to remain distinct.”

What it means to contrast in P is defined as follows.

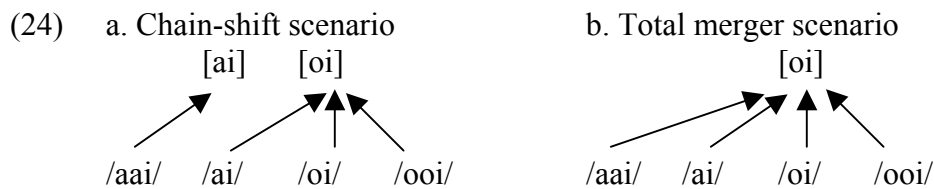
- (23) Def. of Contrast in P
A pair of inputs, in_a and in_b , contrast in P , when corresponding segments in those inputs, seg_a and seg_b , are such that seg_a has P and seg_b lacks P .

P is a potentially contrastive phonological property, such as a distinctive feature, length, stress, presence vs. absence of a segment. The properties P , then, are essentially the same as the properties governed by faithfulness constraints in standard OT. Indeed, $PC(P)$ constraints are like faithfulness constraints in that they look at two levels of representation. But they are novel in that they evaluate contrasts for pairs of underlying words and corresponding output words instead of evaluating individual input-output mappings.

Input-oriented PC constraints, unlike standard faithfulness, admit contrast transformation. Since contrasts can be expressed by various properties, $PC_{IN}(P)$ constraints are satisfied even when contrasts are expressed on the surface in a different way than in the underlying form. In Finnish, for example, even though words that contrast underlyingly in length contrast on the surface in rounding, the $PC_{IN}(\text{long})$ constraint is satisfied. As will be discussed below, $PC_{IN}(P)$ constraints by themselves do

not determine how to preserve particular contrasts. The way in which a given underlying contrast is expressed on the surface is determined by the interaction of input-oriented PC constraints with each other and with other constraints in the theory.

Another role of input-oriented PC constraints is to minimize the number of mergers in a scenario. Given two scenarios that merge the same types of contrast, they prefer a scenario where fewer input pairs are involved in the same type of merger. Compare the chain-shift scenario to a competing total merger scenario. Both merge length and rounding but the total merger scenario merges those properties for more input pairs and thus is non-optimal on input-oriented PC.



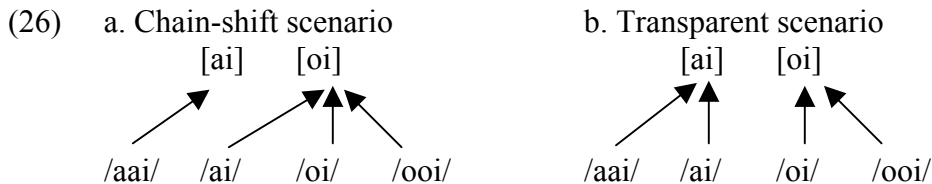
As far as length mergers are concerned, the chain-shift scenario merges two input pairs, $\{/oi/, /ooi/\}$, $\{/ai/, /ooi/\}$. The corresponding total merger scenario, on the other hand, merges four input pairs distinct in length, $\{/oi/, /ooi/\}$, $\{/ai/, /ooi/\}$, $\{/ai/, /aai/\}$, $\{/oi/, /aai/\}$. The same goes for rounding. In the chain-shift scenario, there are two input pairs that merge rounding, $\{/ai/, /oi/\}$, $\{/ai/, /ooi/\}$. In the total merger scenario, rounding is neutralized for four input pairs, $\{/ai/, /oi/\}$, $\{/ai/, /ooi/\}$, $\{/aai/, /oi/\}$, $\{/aai/, /ooi/\}$. Thus, the total merger scenario would never win on input-oriented PC constraints over the chain-shift scenario since it incurs more mergers of each type. But the total merger scenario contains only one output and thus output well-formedness would select it over the competing chain-shift scenario (see chapter 2 for discussion).

1.2.2.1.2 Output-oriented PC

In addition to input mergers, scenarios are evaluated for the ambiguity of their outputs. A scenario with fewer ambiguous outputs is preferred, all else being equal. This is the role of output-oriented PC, $PC_{OUT}(P)$, as defined below.

- (25) $PC_{OUT}(P)$
 For each output that corresponds to two or more inputs contrasting in P assign a violation mark. Formally, assign one mark for every output, out_k , if $in_a \rightarrow out_k$, $in_b \rightarrow out_k$, in_a has P , and in_b lacks P .
 “Avoid outputs ambiguous in P property.”

The primary role of output-oriented PC is to ensure that if mergers take place in a scenario they are accumulated in one location rather than distributed among outputs. This often forces a merger along some additional dimension of contrast. Compare the chain-shift scenario to a transparent scenario.



The two scenarios differ in the distribution of length neutralizations among outputs. The chain-shift scenario contains one output ambiguous in length, the [oi] output. The transparent scenario contains two such outputs, [ai] and [oi]. Thus, in the chain-shift scenario there are fewer outputs that correspond to inputs distinct in length. This is at the cost of merging rounding. There are no rounding neutralizations in the transparent scenario but there are some in the chain-shift scenario. In the chain-shift scenario two pairs of inputs merge in rounding: $\{/ai/,/oi/\}$ and $\{/ai/,/ooi/\}$.

In Finnish, $PC_{OUT}(\text{long})$ ranked above $PC_{IN/OUT}(\text{round})$ selects the chain-shift

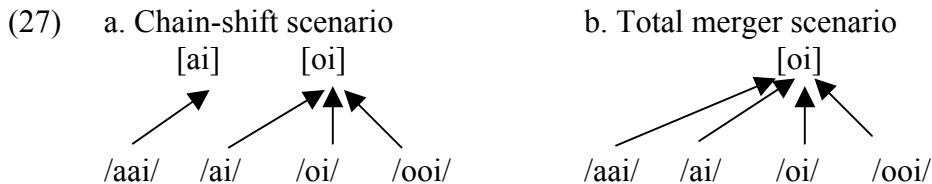
scenario over the transparent scenario since the chain-shift scenario contains fewer outputs ambiguous in length. When ranked higher than PC constraints against rounding mergers, this constraint forces a merger in rounding. As a result, length neutralizations are accumulated in one location in this scenario rather than distributed among outputs.

When neutralizations cluster, there are fewer outputs in a scenario that are ambiguous in some property P. If the number of ambiguous outputs is taken to be an indication of the recoverability of a scenario, output-oriented PC constraints increase recoverability. See section 1.4 for discussion. For previous work on recoverability see Gussmann (1976), Kaye (1974), (1975), Kisseberth (1976). See also evidence for clustering of faithfulness violations in the work of Burzio (1996), (1998). Predecessors of output-oriented PC constraints include output-oriented IDENT-type constraints (see Keer 2000, Struijke 2001).

1.2.2.1.3 Input- and Output-oriented PC

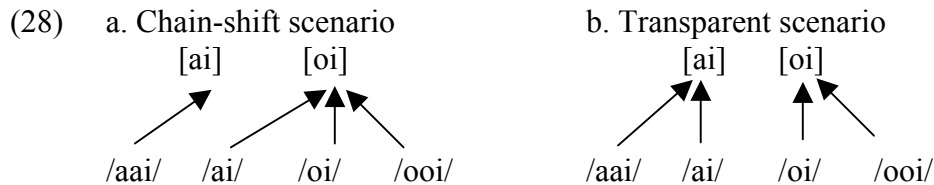
In many cases, input- and output-oriented PC constraints play the same role. They both prohibit neutralizations of particular contrasts. Thus, they are partially overlapping. The difference between the two lies in whether they count neutralizations from the input or the output. This is an important difference when comparing scenarios.

Consider two scenarios with length and rounding mergers, the same number of ambiguous outputs, but a different number of input mergers.



To compare the two scenarios, we need input-oriented PC. Output-oriented PC does not see the difference since it does not count the number of inputs involved in neutralizations. For output-oriented PC, the two scenarios are the same.

Conversely, output-oriented PC is needed to see the virtues of the chain-shift scenario over a transparent scenario.



Input-oriented PC would always prefer the transparent scenario since it avoids merging along one additional dimension of contrast. In the transparent scenario, there are only length mergers. In the opaque scenario, there are both length and rounding mergers. In terms of constraints, $PC_{IN}(\text{long})$ constraint is violated twice in both scenarios. $PC_{IN}(\text{round})$ constraint is violated only in the opaque scenario. Thus, the opaque scenario is harmonically-bounded by the transparent scenario on input-oriented PC constraints. To ensure that the opaque scenario has a chance to win, we need output-oriented PC constraints. $PC_{OUT}(\text{long})$ prefers the opaque scenario since it reduces the number of outputs ambiguous in length.

The output-oriented PC constraint redistributes neutralizations in a scenario onto one output. As a result, there are fewer outputs ambiguous in some property P.

1.2.2.1.4 Relational PC

In PC theory, there are also relational PC constraints. The logic behind relational PC is as follows: even though in chain shifts (and other opaque processes) some contrast

is preserved at the cost of neutralizing some other contrast - in Finnish length is preserved at the cost of neutralizing rounding - there are limits on how many instances of the rounding contrast are neutralized. Too much neutralization may result in an output contrast that does not reflect any minimal instances of an identical input contrast. In Finnish, for example, transforming too many instances of the length contrast into the rounding contrast may result in an output rounding contrast that does not correspond to any instances of the minimal rounding contrast from the input. We can go even further and say that when this happens, the identity of the output contrast is non-recoverable. The output contrast bears no relation to its source. Relational PC militates against it.

Relational PC is related to Kiparsky's Alternation Condition, which prohibits absolute neutralizations. The Alternation Condition bans positing underlying oppositions that are always neutralized on the surface. The following formulation is given by Kenstowicz and Kisseberth (1979, p. 215).

(29) Alternation Condition (Kiparsky 1971)

Each language has an inventory of segments appearing in underlying representations. Call these segments phonemes. The U(nderlying) R(epresentation) of a morpheme may not contain a phoneme /x/ that is always realized phonetically as identical to the realization of some other phoneme /y/.

In short, the Alternation Condition prohibits positing an opposition /x/ vs. /y/ that is always neutralized on the surface.

Similar to the Alternation Condition, relational PC guards identity between output contrasts and their input correspondents. It demands that a given output contrast correspond to at least one instance of an identical minimal input contrast. Relational PC is defined below:

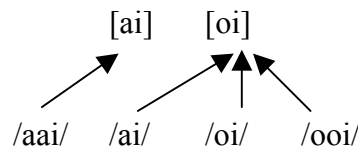
(30) $PC_{REL}(P)$

For a pair of outputs minimally contrasting in P that does not correspond to a pair of inputs minimally contrasting in P , assign a violation mark. Formally, assign one mark for every pair of outputs, out_a and out_b , $|out_a - out_b| = P$, if there is no pair of inputs, in_i and in_j , $\{in_i, in_j\} \rightarrow \{out_a, out_b\}$, and $|in_i - in_j| = P$.

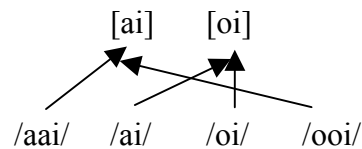
Too much transformation violates relational PC, and thus a conflict results between relational PC and contrast transformation. Given two scenarios that transform contrasts, a scenario in which contrast relation is not preserved loses on relational PC. (See chapter 2.)⁵

Compare the chain-shift scenario as in (31) to a competing bi-directional scenario (bi-directional since it contains movement going in opposite directions). In both, the length contrast is preserved at the cost of neutralizing rounding. In the chain-shift scenario, the length contrast is preserved for two pairs of inputs and realized as a surface contrast in rounding, both /aai/ vs. /ai/ and /aai/ vs. /oi/ map onto [ai] vs. [oi]. The two other input-length contrasts in this scenario, /ooi/ vs. /oi/ and /ooi/ vs. /ai/, are neutralized. In the bi-directional scenario, on the other hand, length is preserved for every pair of inputs and realized as a surface contrast in rounding, both /aai/ vs. /ai/, /ooi/ vs. /oi/, /aai/ vs. /oi/, and /ooi/ vs. /ai/ are realized as [ai] vs. [oi].

(31) a. Chain-shift scenario



b. Bi-directional scenario

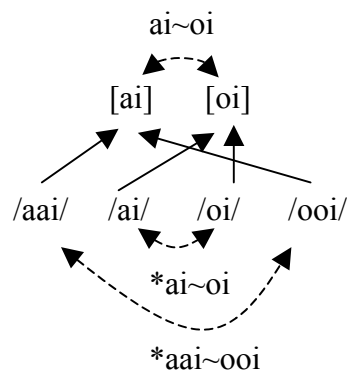


⁵ Relational PC reverses the Alternation Condition. In the Alternation Condition, if /x/ is not equal to /y/ on P , then somewhere [x] is not equal to [y] on P . But in (30), if [x] is not equal to [y] on P , then /x/ is not equal to /y/ on P .

Given the high-preference for preserving length contrasts in Finnish, the bi-directional scenario is expected to come out optimal. It preserves every length contrast from the input.

However, as a result of preserving length for each pair of inputs, in the bi-directional scenario each minimal rounding contrast from the input is neutralized: /aai/ vs. /ooi/ and /ai/ vs. /oi/ map onto [ai] and [oi], respectively. As a result, in the bi-directional scenario, the output rounding contrast does not correspond to any of the minimal instances of the rounding contrast from the input. As already mentioned, when this happens, the identity of the output contrast is non-recoverable. This is illustrated in the following diagram. The diagram indicates minimal rounding contrasts from the input. Contrasts that are neutralized are indicated with an asterisk.

(32) Bi-directional scenario



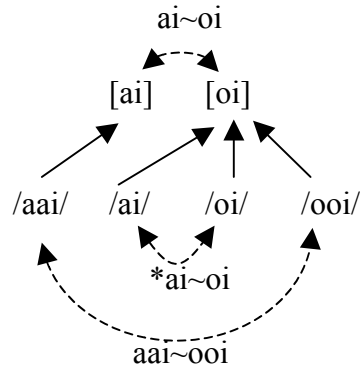
Non-recoverable identity

In the bi-directional scenario, the identity of the output rounding contrast is non-recoverable since neither /aai/ vs. /ooi/ nor /ai/ vs. /oi/ are preserved. Graphically, both contrasts get an asterisk.

In the chain-shift scenario, on the other hand, the output rounding contrast corresponds to one minimal instance of the rounding contrast from the input. While one of the minimal rounding pairs neutralizes - /ai/ vs. /oi/ both map onto [oi] - the other pair,

/aai/ vs. /ooi/, maps onto distinct outputs, [ai] vs. [oi]. Thus, in the chain-shift scenario, unlike in the bi-directional scenario, the identity of the output rounding contrast is recoverable. A minimal rounding contrast from the input, /aai/ vs. /ooi/, is preserved.

(33) Chain-shift scenario



Recoverable identity

Graphically, only one contrast, /ai/ vs. /oi/, gets an asterisk.

In Finnish, the high-ranking $PC_{REL}(\text{round})$ constraint chooses the chain-shift scenario over the bi-directional scenario, since the chain-shift scenario retains some identity between its output and input rounding contrasts. The choice is made in favor of the chain-shift scenario, even though it is the bi-directional scenario that avoids length mergers altogether. Thus, in Finnish, relational PC is in conflict with PC constraints against length mergers. The choice is made in favor of relational PC.⁶

1.2.2.2 Tokenized Markedness

In addition to PC constraints, there are also markedness constraints in the theory. Markedness constraints are indispensable to ignite a shift. As will become apparent, without high-ranking markedness, there would be no movement in a scenario since PC constraints themselves cannot initiate movement (see chapter 4 for discussion). By movement, I mean any unfaithful mapping in a scenario.

In standard OT, markedness constraints evaluate output well-formedness. The same role of markedness is retained in PC theory. In Finnish, for example, high-ranking markedness against tri-moraic syllables accounts for shortening. It is more important to avoid tri-moraic syllables than to preserve length contrasts. But in PC the concept of markedness is taken a step further. Scenarios are different not only on output types but also on how many outputs of a particular type there are in a scenario. (The number of outputs equals the number of inputs that map onto them.) Since markedness in PC counts the number of output types, it is called tokenized markedness.

(34) TOKENIZED MARKEDNESS

Assign a violation mark for every instance of output, out_x , where the number of outputs equals the number of inputs that map onto out_x .

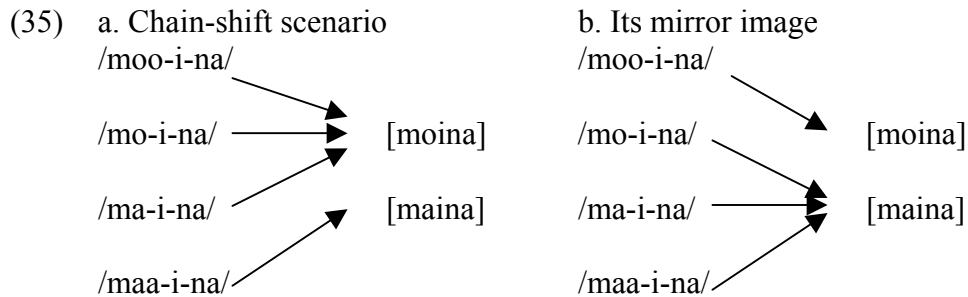
“Assign a violation mark for every token of a marked output in a scenario, where the number of tokens equals the number of inputs that map onto this output.”

When there is no output x in a scenario, tokenized markedness is satisfied. When there is an output x , tokenized markedness is violated and it distinguishes between scenarios with different number of inputs that map onto x . Since there are markedness constraints against various types of outputs and they are ranked with respect to one another, tokenized markedness constraints decide which output in a scenario is a preferred site for neutralization. The less marked output is the one that is preferred according to markedness. It is better to have more instances of a less marked output in a scenario than of its more marked competitor.

Consider two competing chain-shift scenarios. The two scenarios have the same set of outputs but differ on which output is the site of neutralization. In the chain-shift scenario (35a), the output with the [oi] diphthong is the site of neutralization, [moina]. In

⁶ As will be shown in chapter 2, bi-directional contrasts do actually arise.

the mirror-image scenario (35b), it is the output with the [ai] diphthong, [maina]. The scenarios are represented vertically to better illustrate the difference.



Tokenized markedness makes the choice between the two scenarios. The scenario with neutralization onto [moina], the chain-shift scenario, creates more outputs of type [oi]. The scenario with neutralization onto [maina], the mirror-image scenario, contains more tokens of type [ai]. Depending on the relative ranking of tokenized markedness constraints, *ai versus *oi, one or the other scenario is more harmonic. If [maina] is a better output than [moina], then the chain-shift scenario is more harmonic. With the opposite ranking, its mirror image wins.

The two scenarios in (35) differ in directionality of movement. The chain-shift scenario contains rounding, the competing mirror-image scenario contains lowering. Tokenized markedness makes the choice between the two.⁷

To conclude, tokenized markedness constraints have two roles in the theory: (i) they force movement in a scenario by evaluating output well-formedness, and (ii) they evaluate directionality of movement by counting the number of marked tokens in a scenario. Thus, tokenized markedness constraints combine the notions of standard

⁷ Even though tokenized markedness makes a choice between the two types of scenarios, I will argue that in Finnish the *ai markedness constraint is ranked below conflicting PC(round), since it does not force rounding in Finnish. In Finnish, rounding only takes place where shortening occurs. Thus, I will argue that it is due to a high-ranking PC_{OUT}(long) constraint and not *ai markedness.

markedness and faithfulness. They are like standard markedness since they evaluate output well-formedness. But they are also like standard faithfulness since they have access to the input as well as to the output. Consequently, like standard markedness, they can force a phonological process and like standard faithfulness, they can determine directionality of movement in a scenario.⁸

1.2.2.3 Generalized Faithfulness

Markedness and PC constraints constitute the core of PC theory. They belong to stage 1 of Eval and perform the initial screening of scenarios in a candidate set. However, they do not make all necessary distinctions between scenarios. Therefore, once markedness and PC get a chance to trim down the candidate set, faithfulness comes into play. In PC faithfulness constraints belong to stage 2 of Eval. Their role is solely to resolve ties from stage 1. As will become apparent, this is a much more reduced version of faithfulness than the one in standard OT. The reduction of faithfulness is expected in the light of the transfer of much of its responsibility to PC constraints and tokenized markedness.

In PC theory faithfulness has two goals: (i) it rules out scenarios that involve unnecessary movement (movement that is not motivated by either PC or markedness), and (ii) it determines the directionality of movement in a scenario. Movement in PC theory is understood as any type of input-output disparity for each mapping in a scenario. In the following discussion, I will first describe how faithfulness rules out unnecessary movement and then show how it determines the directionality of movement in a scenario.

⁸ Tokenized markedness makes the choice in directionality of movement for most scenarios, except the ones that contain the same number of outputs of each type; in this case generalized faithfulness makes the choice. Bi-directional scenarios are an example. See next section and chapter 2.

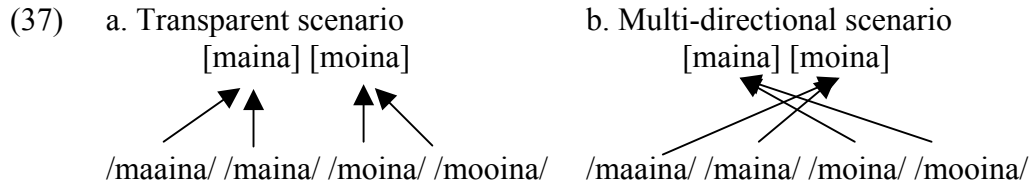
Faithfulness constraints in PC theory are not the same as faithfulness constraints in standard OT. Faithfulness constraints in PC sum up violations of identity for all mappings in a scenario and they treat any change - rounding, nasalization, deletion or insertion the same way. Since they do not distinguish between different types of non-identity, they are called *generalized* faithfulness constraints. This is the main difference between generalized faithfulness and standard faithfulness.

Standard faithfulness distinguishes between different types of non-identity. There are distinct standard faithfulness constraints that militate against a change in rounding, IDENT(round), and different faithfulness constraints that militate against a change in nasality, IDENT(nasal). In PC theory, these are expressed under one and the same constraint. The role of generalized faithfulness, therefore, is to evaluate the cost of achieving a certain interplay of contrasts and markedness in the system. Given a choice between scenarios that fare equally on markedness and PC, generalized faithfulness constraints choose a scenario where inputs and outputs are most similar to each other.

Below I give a preliminary definition of generalized faithfulness. This definition will be modified once the other goal of faithfulness, directionality of movement, is discussed.

- (36) GENERALIZED FAITHFULNESS (first pass)
An output is identical to its input correspondent in every property. Assign a violation mark for any type of disparity (i.e., featural change, deletion, insertion).

Consider two scenarios which are alike except for the degree of input-output disparity. The two scenarios are the transparent scenario with shortening (37a) and multi-directional scenario with shortening, lowering and rounding (37b).



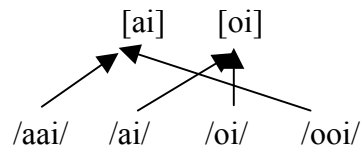
The two scenarios fare the same on markedness and PC. They contain the same outputs and the same number of them (tokenized markedness). They merge the same type and number of input pairs (input-oriented PC). They also both contain the same two ambiguous outputs (output-oriented PC). But the two scenarios are different and the theory should be able to express this formally. This is where generalized faithfulness comes into play.

The two scenarios differ in the degree of disparity between inputs and their corresponding outputs. In the multi-directional scenario, outputs and corresponding inputs are more distant than in the competing transparent scenario. The choice between the two scenarios is left to faithfulness. Faithfulness rules in favor of the transparent scenario since it contains less movement.

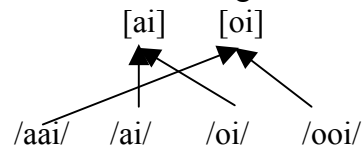
To distinguish between the transparent and multi-directional scenarios, it would be enough for faithfulness to be formulated as a general constraint against input-output disparity. This constraint would then sum up violations of disparity for each mapping in a scenario. However, as the following example shows, a more detailed formulation of the constraint is needed.

Consider two bi-directional scenarios. They are called bi-directional since they contain both rounding (raising) and lowering.

(38) a. Bi-directional scenario



b. Its mirror image



The two scenarios tie on PC and markedness. As far as PC constraints are concerned, both scenarios incur the same types of mergers and involve the same number of inputs and outputs in each merger. As far as markedness goes, both scenarios contain the same outputs and the same number of them. The two scenarios also contain the same degree of input-output disparity since they change length and rounding to the same degree. But the two scenarios are clearly distinct. I will propose that it is the role of generalized faithfulness to distinguish between them.

Let us consider how the two scenarios can be distinguished. Both scenarios contain the same phonological processes: rounding, lowering and shortening. Both scenarios have the same outputs and merge the same inputs. They differ however on what the long and short vowels map onto. In (38a), the long vowels map onto the unrounded vowel output. In (38b), the long vowels map onto the rounded vowel output. Since long vowels shorten, the mapping involving long vowels is less faithful than a comparable mapping involving short vowels ($/aai/ \rightarrow [ai]$ is less faithful than $/ai/ \rightarrow [ai]$). Thus, the two scenarios differ in the distribution of unfaithful mappings across outputs. In (38a), it is the unrounded vowel output that is less faithful. In (38b), it is the rounded vowel output.

To capture this observation formally, I propose that in addition to militating against unnecessary input-output disparity, generalized faithfulness constraints evaluate

disparity separately for different types of outputs in a scenario (the type of output is determined by the presence/absence of the P property). Since faithfulness is evaluated with respect to the type of output, formally, it takes the output segment as its modifier. One can think of it as partitioning the space of outputs into P sets (divided by the type of output) and then evaluating disparity for each mapping in a set. The constraint is defined below.

- (39) [α P]-FAITH (cf. (36))
An [α P] output is identical to its input correspondent in every property. Assign a violation mark for any type of disparity (i.e., featural change, deletion, insertion).

The key idea is that some sets of P properties tend to be more resistant to unfaithful mappings than others, and thus faithfulness, subcategorized to a set, can determine which types of segment(s) can receive a mapping.

Thus, generalized faithfulness that belongs to stage 2 of Eval has two goals: (i) it minimizes input-output disparity, and (ii) determines directionality of movement in cases when the choice is not already made on tokenized markedness. (Tokenized markedness which determines directionality of movement in stage 1 of Eval would not be able to distinguish between the two bi-directional scenarios. Both scenarios fare the same on tokenized markedness: they have the same outputs and the same number of them.)

Let us now evaluate the two bi-directional scenarios with respect to the constraint defined in (39). The two scenarios differ on where long and short vowels map. In (38a), long vowels map onto [ai], the output with the unrounded vowel. In (38b), they map onto [oi], its rounded counterpart. This has consequences for generalized faithfulness. In the scenario where long vowels map onto [ai] (38a), it is the unrounded vowel [ai] that is less faithful, whereas in the other scenario, (38b), it is the rounded counterpart [oi] that fares

worse on faithfulness. Depending on the relative ranking of generalized faithfulness constraints, *unrounded*-FAITH versus *rounded*-FAITH, one or the other scenario comes out optimal.

The generalized faithfulness constraint can be seen as a combination of faithfulness and markedness. It is a faithfulness constraint since it evaluates input-output identity but it is also subcategorized to a particular output and in that, resembles standard markedness. The idea behind generalized faithfulness is that in a given language some outputs are more faithful than other outputs.⁹

The next section shows how PC constraints interact with each other, conflicting markedness constraints, and low-ranked faithfulness in choosing the optimal scenario.

1.3 Illustration of the Proposal

Here is an example of how the constraints work on a simple case of neutralization and the lack of it.¹⁰

1.3.1 Neutralization: Final Devoicing

Consider a language with final devoicing. For final devoicing to take place, markedness against voiced obstruents syllable-finally must outrank conflicting PC. It is more important to avoid voiced obstruents syllable-finally than it is to preserve contrast in voicing. The neutralization ranking is given below.

(40) Neutralization ranking

⁹ One can think of cases where certain phonological alternations take place in the language but some sets of segments do not undergo them. This can be accounted for by generalized faithfulness constraints outranking process-specific markedness constraints.

¹⁰ In the following discussion, there will be no need for the distinction between input- and output-oriented PC nor will it be necessary to count tokens for markedness or subdivide outputs for faithfulness. This is because the following scenarios contain only two mappings. However, for continuity in the presentation of the argument, I will use all the types of constraints introduced so far. It is important to become familiar with them since they will become necessary once the scenarios get larger (see section 1.4).

*VOICEDOBSTRUENT]_σ >> PC_{IN}(voice), PC_{OUT}(voice)

This is illustrated in the following tableau. Scenarios are formed as described in section 1.2.1. For a given three-segment underlying form “scenario-inputs” are all strings of length 0...7. These are all possible combinations of all P (phonological) properties including deletion and insertion of segments. For example, scenario-inputs for underlying form *vad* contain, among others, *vad* (the identity form), *vat* (by final devoicing), *va* (by deletion), *va:* (by deletion & lengthening), *ba* (by deletion & change in place), *thad* (by change in place). The inputs shown in the tableau are a subset of those. They are minimally distinct in obstruent voicing word-finally, *vad* vs. *vat* (This is a standard procedure in OT, where not all forms are shown in a tableau.)¹¹ In each scenario outputs are a subset (possibly improper) of the inputs. In scenario A, the set of outputs is a proper subset of the input. In scenarios B and C, the set of outputs is identical to the set of inputs. We now come to the evaluation.

(41) Final devoicing takes place

Scenarios		*Voi Obs] _σ	PC _{IN} (voice)	PC _{OUT} (voice)	(+vd)- FAITH	(-vd)- FAITH
A. Neutralization (Polish)	/vad/ → vat /vat/ → vat		* {/vad/,/vat/}	* [vat]		* d→t
B. Identity (English)	/vad/ → vad /vat/ → vat	*!				
C. Permuted (Not attested)	/vad/ → vat /vat/ → vad	*!			* t→d	* d→t

¹¹ Another way to look at it is to let arguments of relevant PC constraints, here PC(voice), choose what input contrasts are shown in a tableau.

Scenario A, the neutralization scenario, is the winner as it satisfies high-ranked markedness. It contains no outputs with a voiced obstruent syllable-finally. The other two scenarios, scenario B and scenario C, violate high-ranked markedness since they contain a form with a voiced obstruent in a word-final position. The neutralization scenario, scenario A, incurs some PC violations - it merges contrast for one pair of inputs (violation of input-oriented PC) and in so doing creates an ambiguous output (violation of output-oriented PC), but PC constraints are low-ranked and so are less important than high-ranked markedness. Faithfulness violations are irrelevant here since the choice between candidates is already made in stage 1 of Eval, even before faithfulness gets a chance to apply. There is one violation of faithfulness in the actual scenario since there is voicing disparity in one of the mappings. There are two violations in the permuted scenario, scenario C, since there is voicing disparity in both mappings. (The permuted scenario will be discussed in more detail in the following section.)

1.3.2 Lack of Neutralization

Now consider the lack of neutralization. In this case, the underlying voicing contrast is preserved on the surface. In terms of a constraint ranking, PC dominates conflicting markedness. It is more important to preserve contrast than to avoid forms that violate markedness. The relevant ranking is below:

- (42) Lack of neutralization
 $PC_{IN}(\text{voice}), PC_{OUT}(\text{voice}) \gg *VOICEDOBSTRUENT]_{\sigma}$

This is illustrated in the following tableau. Scenarios are formed as in the previous case.¹²

- (43) No devoicing

Scenarios		PC_{IN} (voice)	PC_{OUT} (voice)	*Voi Obs] $_{\sigma}$	(+vd)- FAITH	(-vd)- FAITH
A. Neutralization (Polish)	/nid/ → nit /nit/ → nit	* {/nid/,/nit/}	*! [nit]			* d→t
B. Identity (English)	/nid/ → nid /nit/ → nit			*		
C. Permuted (Not attested)	/nid/ → nit /nit/ → nid			*	* t→d	*! d→t

Here the identity scenario, scenario B, is the winner as it satisfies high-ranked PC. In this scenario, forms distinct in voicing are kept distinct even at the cost of violating markedness. The neutralization scenario, scenario A, loses on PC.

The above tableau also illustrates the role of low-ranked FAITHFULNESS. Consider the so-called permuted scenario, candidate C, which is like the identity scenario except

that in this scenario outputs correspond to different inputs. The choice between candidate B, the identity scenario, and candidate C, the permuted scenario, cannot be made on PC or markedness. The two fare the same on those two types of constraints. This is where we need faithfulness. Faithfulness favors candidate B (the identity scenario) over candidate C (the permuted scenario). In the identity scenario, outputs are closer to their inputs than in the competing permuted scenario (in fact they are the same), and this is preferred, all else being equal.

This illustrates an important prediction of PC theory that is different from rule-based approaches but similar to standard OT. In rule-based approaches mappings take place as long as there exist rules of a particular type. In standard OT, mappings are more restricted. They take place only when they improve on markedness (Moreton 1997). In PC theory, similarly, generalized faithfulness rules out unnecessary movement. For movement to take place, it must improve on either contrast or markedness. Otherwise, it will not take place. (For a discussion of this prediction see chapter 4.)

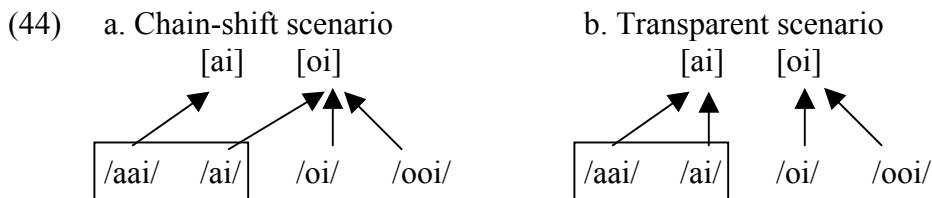
So far, PC works similarly to standard faithfulness. When markedness dominates conflicting PC, a phonological process takes place. When PC dominates conflicting markedness, a phonological process is blocked. The next section points to differences between PC and standard faithfulness.

¹² In this example, the ranking is provable for only one PC constraint. This is expected, since the two PC constraints, input and output, both militate against neutralization, and thus if neutralization takes place, they are both violated. However, it will become clear in chapter 2 that both constraints are required. It is necessary to retain both of them for an opaque scenario to ever win over other scenarios in a candidate set. If there were no input-oriented PC constraints, the so-called total merger scenario would always win over competing chain-shift scenario (or any other competing opaque scenario for that matter). If there were no output-oriented PC constraints, the so-called transparent scenario would always win over the chain-shift scenario.

1.4 Application to Chain Shifts: Contrast Transformation and Neutralization

As has been observed, in Finnish the underlying length contrast is preserved despite shortening and realized as surface contrast in rounding. Some instances of the original rounding contrast are neutralized as a result. This is called *contrast transformation*. (A complete set of vowel alternations in Finnish is discussed in chapter 3.)

Compare the chain-shift scenario to a competing transparent scenario. In both scenarios there is shortening but only the chain-shift scenario involves rounding. In the chain-shift scenario, due to rounding, the length contrast is preserved for one minimal pair of inputs despite shortening. The length contrast is realized as a rounding contrast, /aai/ vs. /ai/ is manifested as [ai] vs. [oi]. In the transparent scenario, on the other hand, there is no rounding, and thus the two inputs, /aai/ vs. /ai/, map onto the same output. The relevant input pair is boxed.

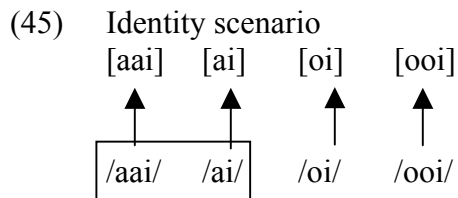


Shortening neutralizes all minimal length contrasts in the transparent scenario but is non-neutralizing for some input contrasts in the chain-shift scenario. Preservation of the length contrast versus its neutralization is the crucial difference between the chain-shift scenario and a competing transparent scenario.

Another way to look at the difference between the chain-shift scenario and the transparent scenario is in terms of output ambiguity. As was discussed in previous sections, rounding in the chain-shift scenario in comparison to the transparent scenario

reduces the number of outputs that correspond to inputs distinct in length. In the chain-shift scenario there is only one output ambiguous in length, the [oi] output. In the transparent scenario, on the other hand, there are two such outputs, [ai] and [oi].

In the following discussion, I will first explain why shortening takes place and then account for rounding. Let us compare the chain-shift scenario to the identity scenario shown below. The identity scenario does not merge any contrasts, thus wins on PC constraints, but it contains long vowels in the output.¹³



For the chain-shift scenario to win over the identity scenario, it must be essential for long vowels to shorten.

I propose that in Finnish shortening takes place to avoid tri-moraic syllables. Some length contrasts are neutralized as a result. In terms of a constraint ranking, a markedness constraint against tri-moraic syllables, $*\sigma_{\mu\mu\mu}$, must outrank conflicting PC constraints against length mergers, $PC_{OUT}(long)$ and $PC_{IN}(long)$. The relevant markedness constraint and the constraint ranking that accounts for shortening are given below.

¹³ In reality, each scenario merges some contrasts due to ROTB, but the identity scenario does not merge length or rounding contrasts for the relevant inputs.

- (46) a. $*\sigma_{\mu\mu\mu}$
 Do not have syllables of three nuclear moras.
- b. $*\sigma_{\mu\mu\mu} \gg \text{PC}_{\text{OUT}}(\text{long}), \text{PC}_{\text{IN}}(\text{long})$

This is illustrated in the following tableau. For clarity of exposition, in the following tableau I list each violation of a constraint as a star and I also indicate next to it a form or a pair of forms that incurs the violation. Markedness is evaluated for each output form in a scenario, and the forms in square brackets are the outputs that violate a particular markedness constraint. For output-oriented PC, the form in square brackets is the output that corresponds to inputs contrasting in length. For input-oriented PC, the pair in curly brackets is the pair that neutralizes length.

Scenarios in the candidate set are formed as described in section 1.2.1. All scenarios in the same candidate set contain the same inputs. Scenario-inputs are generated by a function Gen (similar to the role of Gen in correspondence theory). For an underlying form *maa-i-na*, and any other form of that length, scenario-inputs contain all strings of length 0...13 (I assume that long vowels count as one segment). Thus, scenario-inputs for that underlying form contain forms, such as, *maa-i-na* (identity form), *ma-i-na* (by shortening), *baa-i-na* (by denasalization), *ba-i-na* (by shortening & denasalization) etc. The forms shown in a tableau are a subset of those. They are minimally contrastive on vowel length and vowel rounding. Outputs in the scenarios are a subset (possibly improper) of the input. In the identity scenario, the forms in the output are identical to the forms in the input. In the actual scenario and in the total merger scenario, they constitute a proper subset of the input.

(47) Shortening takes place

Scenarios		* $\sigma_{\mu\mu\mu}$	PC _{OUT} (long)	PC _{IN} (long)
A. Identity aai \rightarrow ai ooi \rightarrow oi	/maa-i-na/ \rightarrow maa-i-na /ma-i-na/ \rightarrow ma-i-na /moo-i-na/ \rightarrow moo-i-na /mo-i-na/ \rightarrow mo-i-na	**! [aai], [ooi]		
B. Actual aai \rightarrow ai ooi \rightarrow oi	/maa-i-na/ \rightarrow ma-i-na /ma-i-na/ \rightarrow mo-i-na /moo-i-na/ \rightarrow mo-i-na /mo-i-na/ \rightarrow mo-i-na		* [oi]	** {/ai/, /ooi/} {/oi/, /ooi/}
C. Total merger aai \rightarrow ai ooi \rightarrow oi	/maa-i-na/ \rightarrow mo-i-na /ma-i-na/ \rightarrow mo-i-na /moo-i-na/ \rightarrow mo-i-na /mo-i-na/ \rightarrow mo-i-na		* [oi]	****! {/ai/, /ooi/} {/oi/, /ooi/} {/ai/, /aai/} {/oi/, /aai/}

The identity scenario, scenario A, loses on markedness since it contains tri-moraic syllables. The other two scenarios, the actual scenario and the total-merger scenario, both satisfy markedness, but the total-merger scenario incurs too many mergers of length. It merges length for four input pairs. Scenario B is optimal. It merges length but for fewer pairs than the total-merger scenario. In this scenario only two input pairs merge in length.

The ranking so far explains why shortening takes place but it does not account for rounding. Compare the actual scenario to a scenario with shortening but no rounding, the transparent scenario. In both, there is shortening at the cost of neutralizing length. But in the actual scenario there is also rounding and thus some rounding contrasts are neutralized in addition to length contrasts. Neutralizations of length and rounding contrasts in the actual scenario and in the competing transparent scenario are evaluated below.

(48) Input-oriented PC

Scenarios		PC _{IN} (long)	PC _{IN} (round)
A. Actual aai → ai ooi → oi	/maa-i-na/ → ma-i-na /ma-i-na/ → mo-i-na /moo-i-na/ → mo-i-na /mo-i-na/ → mo-i-na	** {/oi/, /ooi/} {/ai/, /ooi/}	** {/ai/, /oi/} {/ai/, /ooi/}
B. Transparent aai → ai ooi → oi	/maa-i-na/ → ma-i-na /ma-i-na/ → ma-i-na /moo-i-na/ → mo-i-na /mo-i-na/ → mo-i-na	** {/aai/, /ai/} {/ooi/, /oi/}	

In both scenarios length is merged for the same number of input pairs, but in the actual scenario in addition there are mergers of rounding. In the following discussion, I will explain what forces rounding in the actual scenario and how it aids length.

Observe that rounding improves on the distribution of length neutralizations in a scenario. In the actual scenario, due to rounding, length mergers are accumulated locally rather than distributed across outputs. As a result, there is an output that does not participate in any length mergers (in fact it does not participate in any merger at all), and thus stands in a bi-unique relation to its input. Thus, rounding reduces the number of ambiguous outputs in a scenario. If we take the number of ambiguous outputs to be an indication of the recoverability of a scenario, rounding improves recoverability (Gussmann 1976, Kaye 1974, 1975, Kisseberth 1976).

For the actual scenario to win, it must be more important to improve on the distribution of length neutralizations in a scenario (contributing to bi-uniqueness) than to avoid merging rounding. This is what forces rounding in Finnish.

(49) PC_{OUT}(long) >> PC_{IN}(round), PC_{OUT}(round)

This is illustrated in the following tableau.

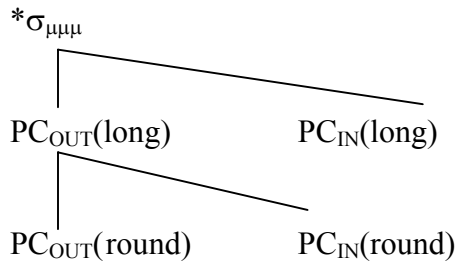
(50) Vowel rounding takes place

Scenarios		* $\sigma_{\mu\mu\mu}$	PC _{IN} (long)	PC _{OUT} (long)	PC _{IN} (round)	PC _{OUT} (round)
A. Actual aai → ai oai → oi	/maa-i-na/ → ma-i-na /ma-i-na/ → mo-i-na /moo-i-na/ → mo-i-na /mo-i-na/ → mo-i-na		** {/oi/, /ooi/} {/ai/, /ooi/}	* [oi]	** {/ai/, /oi/} {/ai/, /ooi/}	* [oi]
B. Transparent aai → ai oai → oi	/maa-i-na/ → ma-i-na /ma-i-na/ → ma-i-na /moo-i-na/ → mo-i-na /mo-i-na/ → mo-i-na		** {/aai/, /ai/} {/ooi/, /oi/}	**! [ai] [oi]		

The actual scenario, with rounding, wins on PC_{OUT}(long). It contains only one output ambiguous in length, whereas the transparent scenario contains two such outputs.

The constraint ranking established in this section is summarized below. Ranking arguments follow.

(51) Constraint ranking



(52) Ranking arguments

Ranking

* $\sigma_{\mu\mu\mu}$ >> PC_{OUT}(long), PC_{IN}(long)

PC_{OUT}(long) >> PC_{OUT}(round), PC_{IN}(round)

Consequence

Long vowels are avoided at the cost of merging length.

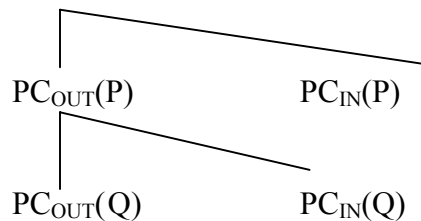
Length neutralizations are accumulated in a scenario by merging rounding.

Recall that shortening takes place to avoid tri-moraic syllables. Due to shortening, some length contrasts are merged ($*\sigma_{\mu\mu\mu} \gg PC_{OUT}(long), PC_{IN}(long)$). But length mergers need to be well-distributed (accumulated locally) in a scenario, and this is at the cost of merging rounding ($PC_{OUT}(long) \gg PC_{OUT}(round), PC_{IN}(round)$). As a result, rounding takes place. Formally, rounding is an indirect consequence of shortening and a high-ranked requirement on the accumulation of length neutralizations in a scenario.

The chain-shift scenario preserves some length contrasts despite shortening. Length contrast is preserved for one minimal pair of inputs in a scenario, /aai/ vs. /ai/, and is manifested as a surface rounding contrast. Thus, *contrast transformation* takes place.¹⁴

Below is the general chain shift schema in cases like Finnish:

(53) Chain shift schema (cf. (51))
*P



P-type segments are avoided due to high-ranked markedness and this results in some P mergers ($*P \gg PC_{OUT}(P), PC_{IN}(P)$). But there is a high-ranking requirement on the distribution of P mergers in a scenario, output-oriented PC. It requires that fewer outputs

¹⁴ To capture this observation formally, one could employ PC constraints that would require preservation of minimal input contrasts only. The problem with such constraints is that they would not assign violation marks to mergers of non-minimal distinctions at all (those, for example, take place in the unattested cross-corner scenario discussed in chapter 2). Therefore, there would be a constraint ranking, under which a scenario that does not merge minimally distinct forms but incurs long-distance mergers wins. This is not a good prediction and that is why I do not pursue this approach further. (Another alternative would be to use both general PC constraints and PC constraints against mergers of minimal contrasts.)

correspond to inputs contrasting in P. That forces Q mergers, as only by merging along some other dimension of contrast can the contrast distribution requirement be satisfied as much as possible ($PC_{OUT}(P) \gg PC_{OUT}(Q), PC_{IN}(Q)$). In effect, some original instances of the P contrast are preserved on the surface and manifested as contrast Q. Some instances of the Q contrast are lost.

This illustrates an important prediction of PC theory. In PC, a phonological process can take place solely to improve on the distribution of some contrast in a scenario. In the schematic example above, P distribution is improved by Q mergers. To improve P distribution means to accumulate neutralizations of P in one location in a scenario rather than distributing them among outputs. As a result, fewer outputs are ambiguous in P. This is the role of output-oriented PC. The high-ranking $PC_{OUT}(P)$ constraint forces a phonological process (the Q process). This shows that PC constraints can activate a phonological process without reference to a high-ranking markedness constraint, as in standard OT. Before further discussion of this prediction, let us look at another example of a chain shift effect.¹⁵

1.5 Another Example: Kashubian Shift

As we have seen, in Finnish shortening forces rounding. Formally, rounding is a consequence of a high-ranked markedness constraint against tri-moraic syllables and a high-ranked PC constraint on the accumulation of length mergers in a scenario. This results in the preservation of some minimal input-length contrasts at the expense of

¹⁵ To explain rounding in Finnish, Harrikari (2000) and Anttila (2000) propose a high-ranking markedness constraint against diphthongs of the form *ai*. Harrikari formalizes it as a constraint against nuclei with maximal contour outside of the head syllable, $*[MAXCONT]_{Nuc(Non-H)}$. In PC theory, though the **ai* constraint is required to make all distinctions between scenarios in the same candidate set (see chapter 2), it is dominated by conflicting PC. Rounding takes place as a result of shortening together with the requirement on preserving contrast (see chapter 4).

rounding. In this section, I will analyze a parallel shift from a north-central Polish dialect, Kashubian. When analyzing Kashubian vowels, I will assume underlying representations of standard Polish. The evidence for this assumption comes from borrowings, some of which surface with standard Polish vowels (Lorentz 1958).

As will be discussed below, in Kashubian the chain shift effect involves processes of vowel fronting and raising. In particular, I will argue that fronting forces raising. The analysis will then proceed in parallel with the analysis of Finnish given in the previous section.

Before presenting the details of the analysis, let us look at the vowel inventory of standard Polish, given below.

(54) Vowel inventory of standard Polish (Rubach 1984)

i ɨ u
 e o
 a

There are six oral vowels in Polish. Using the features of high, low, back and round, the following is the vowel classification, according to Rubach (1984). (SPE style.)

(55) Vowel features in Polish

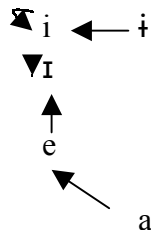
	i	u	ɨ	e	o	a
High	+	+	+	-	-	-
Low	-	-	-	-	-	+
Back	-	+	+	-	+	+
Round	-	+	-	-	+	-

Polish dialects show variation in vowel realization (Dejna 1993, Urbańczyk 1972). Some of the variation has the character of chain shifts. Kashubian shift is an example. The

Kashubian region is itself diverse. In the description below, I will attempt to concentrate on the most prevalent vowel mappings in Kashubian.

In Kashubian, the low vowel undergoes raising and is realized as the mid vowel [e].¹⁶ As a result, the mid vowel [e] raises towards [i]. Sometimes it is realized slightly lower than [i], indicated as raised [ĕ] or [ɪ]. (In some areas it is also realized as [ie], [iy] or even central high vowel [ɨ]). The high vowels in Kashubian, [i] and [ɨ], tend to neutralize, both mapping onto [i].¹⁷

(56) Kashubian vowel shift



Some examples are given below.

(57) Some examples (Dejna 1993, Urbańczyk 1972)

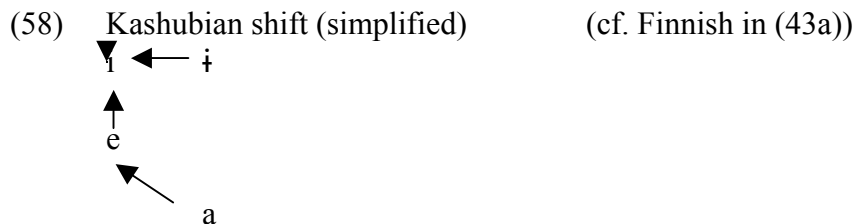
<i>Standard Polish</i>	<i>Kashubian</i>	
	/a/ → [e]	
tr[a]va	tr[e]va	‘grass’
pt[a]k	pt[e]χ	‘bird’
g[a]d[a]	g[e]d[e]	‘he talks’
por[a]nek	por[e]nk	‘morning’

¹⁶ In some regions of Kashubian, as reported by Lorentz (1958), the low vowel raises but it does not make it all the way to [e]. It stops somewhere below [e]. In this section I will discuss raising in the front vowels only. But in some regions the low vowel [a] also surfaces as [o] or even [u]. This then causes the /o/ to [u] raising.

¹⁷ Sometimes both /i/ and /ɨ/ (and sometimes [u]) map onto an unrounded mid vowel [ĕ] ([ɔ]) (not after palatals). Thus, there is total neutralization in the high vowel set. In some sources, it is indicated that the original difference in backness between unrounded high vowels /i/ and /ɨ/ is preserved on the surface by palatalization or lack of it on the preceding consonant.

	/e/ → [ɪ]	
bž[e]k	bž[ɪ]k	‘shore’
s[e]r	s[ɪ]r	‘cheese’
ml[e]ko	ml[ɪ]ko	‘milk’
	/ɨ/ → [i]	
d[ɨ]m	d[i]m	‘smoke’
r[ɨ]ba	r[i]ba	‘fish’
b[ɨ]ć	b[i]ć	‘to be’
v[ɨ]ć	v[i]ć	‘to howl’
	/i/ → [i]	
ń[i]va	ń[i]va	‘soil’
b’[i]ć	b’[i]ć	‘to beat’
v’[i]ć	v’[i]ć	‘to plait’

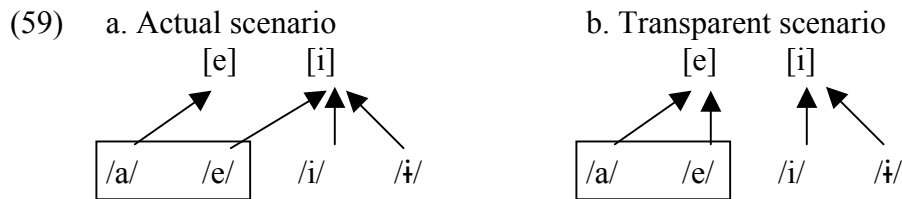
For the purposes of the presentation of the argument, I will assume temporarily that /e/ raises all the way to [i] (which in fact is the case in some other dialects of Polish). Some sources on Kashubian also report this mapping. Once the argument is clear, I will then explain why /e/ stops at [ɪ]. This simplification has no effect on the analysis but allows us to achieve a full parallel with Finnish. Thus, we have the following set of mappings.



Drawing on the parallel with Finnish, I propose that in Kashubian, fronting takes place to avoid central vowels. Both high and low vowels undergo fronting. Vowel raising is then a result of fronting. Specifically, /e/ raises to [i] because /a/ fronts to [e].

Kashubian shift represents *contrast transformation*. If there were no raising of /e/, the vowels /a/ and /e/ would map onto the same output - they would both map onto [e]. However, due to raising, the two vowels contrast on the surface despite fronting, /a/ vs. /e/ are realized as [e] vs. [i]. Descriptively, the underlying contrast in backness is transformed into surface contrast in height.

In terms of output ambiguity, due to raising, there is only one output in the scenario ambiguous with respect to underlying backness, the [i] output. If there were no raising, there would be two such outputs, [i] and [e]. (See the parallel with Finnish discussed in the previous section.) The actual scenario with fronting and raising, and the competing transparent scenario with no raising of /e/ are represented below.



In the actual scenario, mergers in backness are accumulated locally rather than distributed among outputs. This is at the cost of merging height. While in the transparent scenario, there are no inputs that merge height, in the actual scenario, there are two such input pairs, {/e/,/i/} and {/e/,/ɨ/}.¹⁸

For the actual scenario to win, it must be more important to reduce output ambiguity in backness than to avoid neutralizing height. The following ranking accounts for this effect:


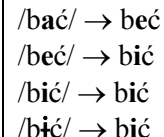
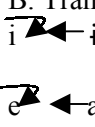
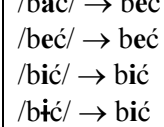
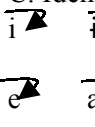
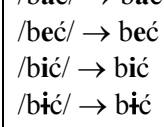
(60) *central >> PC_{OUT}(back) >> PC_{IN}(high), PC_{OUT}(high) (cf. (51))

¹⁸ I assume that /a/ and /e/ do not merge height. They are both [-high].

Informally, e-raising is a result of fronting together with the requirement on reducing output ambiguity in backness.

The constraint ranking is illustrated in the following tableau. The tableau includes the actual chain-shift scenario, the competing transparent scenario with no raising, and the identity scenario.

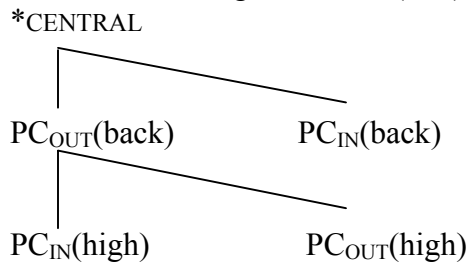
(61) Kashubian

Scenarios		*CENTRAL	PC _{IN} (bk)	PC _{OUT} (bk)	PC _{IN} (high)	PC _{OUT} (high)
A. Actual 			** {/i/, /ɨ/} {/e/, /ɨ/}	* [i]	** {/e/, /i/} {/e/, /ɨ/}	* [i]
B. Transparent 			** {/i/, /ɨ/} {/e/, /a/}	**! [i] [e]		
C. Identity 		**!				

The chain-shift scenario, scenario A, is optimal. It avoids central vowels by merging some distinctions in backness but it reduces the number of outputs ambiguous in backness. There is only one output ambiguous in backness in the actual scenario, scenario A, but two such outputs in the competing transparent scenario, scenario B. The actual scenario reduces the number of outputs ambiguous in backness by means of e-raising. This results in neutralizations of some input height distinctions. In the actual scenario, some vowels distinct in height merge on the surface, while in the corresponding transparent scenario height contrasts are preserved in surface forms.

The constraint ranking is summarized below.

(62) Constraint ranking (cf. (49))



Ranking arguments are recalled below.

(63) Ranking arguments (cf. (51))

Ranking

*CENTRAL >> PC_{OUT}(back), PC_{IN}(back)

Consequence

Central vowels are avoided at the cost of merging backness.

PC_{OUT}(back) >> PC_{OUT}(high), PC_{IN}(high)

Front-back neutralizations are accumulated in a scenario by merging height.

In short, central vowels are avoided and this results in some mergers in backness. But those mergers tend to be accumulated in a scenario onto one output rather than distributed among outputs. This is at the cost of merging height.

Let us now go back to the actual shift and explain why /e/ does not raise all the way to [i] but stops at the lax vowel [ɪ]. I call it the incomplete shift. In the following tableau, the scenario where /e/ stops at [ɪ] (scenario A, the incomplete shift) is compared to scenarios where it raises all the way to [i] (scenarios B and C). Each scenario contains five inputs, one of which is the high lax vowel [ɪ]. In scenario A, the high lax vowel is one of the outputs. In the other two scenarios, it does not surface. In scenario B, it maps onto [i]. In scenario C, it deletes.

(64) Incomplete shift

Scenarios		*CENTRAL	PC _{IN} (bk)	PC _{OUT} (bk)	PC _{IN} (high)	PC _{OUT} (high)	*I
A. Actual i ← ĭ I ▲ e ← a	/bać/ → beć /beć/ → bić /bɪć/ → bić /bić/ → bić /bĩć/ → bić		* {/i/, /ĩ/}	* [i]	* {/e/, /ɪ/}	* [ɪ]	**
B. Raising to [i] i ← ĭ I ▲ e ← a	/bać/ → beć /beć/ → bić /bɪć/ → bić /bić/ → bić /bĩć/ → bić		***! {/i/, /ĩ/} {/e/, /ĩ/} {/ɪ/, /ĩ/}	* [i]	***** {/e/, /ɪ/} {/e/, /i/} {/ɪ/, /i/} {/e/, /ĩ/} {/ɪ/, /ĩ/}	* [i]	
C. Raising to [i] i ← ĭ I ▲ e ← a	/bać/ → beć /beć/ → bić /bɪć/ → ∅ /bić/ → bić /bĩć/ → bić		**! {/i/, /ĩ/} {/e/, /ĩ/}	* [i]	** {/e/, /i/} {/e/, /ĩ/}	* [i]	

Scenario A, the incomplete shift, is optimal since it merges the fewest number of input pairs (in both backness and height). This is reflected in the violation marks for input-oriented PC constraints. In scenario A, there is only one input pair that merges backness and only one that merges height. In the other two scenarios, scenarios B and C, there are more such input pairs. Thus, raising “half way,” as in the actual scenario, reduces the number of input pairs that participate in neutralizations.

In traditional terms, the actual scenario in Kashubian is non-structure-preserving. It contains an output that does not surface elsewhere in the language, the [ɪ] output. A non-structure-preserving scenario, like the one in Kashubian, reduces the number of input mergers in a scenario. This is because it contains more outputs than the competing scenarios in the same candidate set, and thus offers more possibilities for the same inputs

to map onto. At the same time, since it contains more outputs, it incurs more markedness violations.

Therefore, for a non-structure-preserving scenario to win it is more important to improve on input-oriented PC than to improve on markedness. In Kashubian, the actual scenario violates the markedness constraint ***I** since it contains the [ɪ] output. But because ***I** is ranked lower than conflicting input-oriented PC constraints, the non-structure-preserving scenario wins.¹⁹

The [ɪ] vowel in scenarios B and C does not surface. This raises an interesting question: what happens to inputs that never surface? Two possibilities are entertained here, they either delete (as in scenario C) or map onto one of the existing outputs (as in scenario B). In our example, there is no difference between the two alternatives. However, since deletion incurs mergers of the V/∅ type, it may be non-optimal in a language where V/∅ mergers are highly marked.

To conclude, in Kashubian raising is forced by the process of fronting. Formally, it is a result of a high-ranked markedness constraint against central vowels and a high-ranked PC constraint on the accumulation of backness mergers in a scenario. Thus, a parallel is established with the Finnish chain shift discussed in section 4.

1.6 PC as Faithfulness and Markedness

In standard Optimality Theory phonological mappings are accounted for by the relative ranking of markedness and faithfulness constraints. Markedness constraints demand output well-formedness. Faithfulness constraints call for input-output identity.

The two often conflict and their conflict is resolved by constraint ranking. When markedness outranks conflicting faithfulness, a phonological process takes place. With the opposite ranking, a phonological process is blocked.

In PC theory some of the role previously assigned to markedness and faithfulness constraints is taken over by novel PC constraints. As will be explained below, PC constraints infringe on the territory previously assigned to markedness and faithfulness. Since PC constraints take on some of the role of both markedness and faithfulness, they somewhat blur the distinction between the two seemingly distinct families of constraints. Let us consider the dual nature of PC constraints.

PC as Faithfulness. Standard faithfulness requires input-output identity in a particular phonological property. Thus, when ranked above conflicting markedness, faithfulness constraints block a phonological process. Like standard faithfulness, PC constraints block a phonological process when ranked higher than conflicting markedness and allow it to apply with the opposite ranking. This was shown in section 1.3 with the example of final devoicing. When PC against the voicing merger outranked markedness against voiced obstruents syllable-finally, final devoicing was blocked. It was more important to preserve the voicing contrast than to satisfy markedness. When, on the other hand, markedness was ranked higher than the conflicting PC constraint, final devoicing took place. Markedness satisfaction was the top priority. The rankings for neutralization and the lack of it are recalled below.

¹⁹ Structure preservation figures prominently in the work of Kiparsky (1968, 1971). Kiparsky proposes that lexical rules have to be structure preserving. It thus follows that they have to be neutralizing. This goes hand in hand with the generalization of PC theory, by which a structure-preserving scenario incurs more input mergers (thus is more neutralizing) than a competing non-structure-preserving scenario.

- (65) Neutralization and the lack of it
 Final devoicing *VOICEDOBSTRUENT]_σ >> PC_{IN}(voice), PC_{OUT}(voice)
 No devoicing PC_{IN}(voice), PC_{OUT}(voice) >> *VOICEDOBSTRUENT]_σ

Both PC constraints and standard faithfulness ensure contrast preservation but only PC constraints allow for contrast transformation. PC constraints are satisfied even when a given underlying contrast is transformed into a different surface contrast. Standard faithfulness does not allow for that. Thus, if the obstruent voicing contrast is manifested by contrast in vowel length, PC constraints are satisfied but standard faithfulness constraints are violated.

In other words, standard faithfulness constraints do not distinguish between the loss and transformation of a phonological property, treating the two as equal violations of a given faithfulness constraint. PC constraints, in turn, favor transformation to the loss of contrast. One can then ask whether in PC theory contrast transformation is not always better than the loss of contrast. After all, transformation preserves contrast. But transformation has consequences for a system of mappings. Transformation results in additional mergers, some of which may be disallowed in a given grammar. These are some of the differences between PC and standard faithfulness. Let us now move on the discussion of PC as markedness

PC as Markedness. Markedness constraints are constraints on output well-formedness. When ranked higher than conflicting faithfulness, they force a phonological process. PC constraints take on some of the role of markedness, in that they are able to activate a phonological process. In section 1.4 we have seen an example of a Finnish chain shift, where the latter process in the shift, the process of rounding, is forced by a

high-ranking PC constraint on the accumulation of length mergers in a scenario. In section 1.5 we have seen an example of a Kashubian shift where raising is forced by a PC constraint on the accumulation of backness mergers in a scenario. The relevant rankings are recalled below.

- (66) Chain shift rankings
- | | |
|-----------|---|
| Finnish | $*\sigma_{\mu\mu\mu} \gg \text{PC}_{\text{OUT}}(\text{long}) \gg \text{PC}_{\text{OUT}}(\text{round}), \text{PC}_{\text{IN}}(\text{round})$ |
| Kashubian | $*\text{CENTRAL} \gg \text{PC}_{\text{OUT}}(\text{back}) \gg \text{PC}_{\text{OUT}}(\text{high}), \text{PC}_{\text{IN}}(\text{high})$ |

Unlike standard markedness, PC constraints can only activate a phonological process when there is another mapping that takes place in the system. In a chain-shift scenario, PC constraints can only force the latter mapping in the shift. The initial mapping has to take place for reasons of output well-formedness.

Thus we have seen that PC constraints act like both markedness and faithfulness constraints in standard OT. They combine the properties of the two previously distinct forces in the grammatical system. This double identity of PC constraints follows from the architecture of PC theory and allows us to provide a uniform explanation of opaque (chain shifts) and transparent processes. Opaque processes are such that they cannot always be accounted for by markedness ranking. PC accounts for them.

It is important to note that PC constraints do not simply replace markedness and faithfulness constraints from standard OT, but they take on some of the role previously assigned to the two families of constraints. It is indispensable, however, to retain separate markedness and faithfulness constraints (though the constraints are formulated in slightly different terms than in standard OT).

Generalized Faithfulness. In PC theory, faithfulness constraints are required to rule out unnecessary movement. This becomes relevant when two scenarios tie on PC

and markedness, but one of the scenarios involves too much disparity between its inputs and corresponding outputs. Faithfulness then rules in favor of the scenario where outputs are overall closer to their inputs. Generalized faithfulness is also necessary to determine directionality of movement in a scenario.

Tokenized Markedness. Markedness constraints in PC are indispensable to ignite a shift. As we have seen, in Finnish, if there were no high-ranking markedness against tri-moraic syllables, there would be no shift – no shortening and thus no rounding. PC by itself can force the latter step in the shift, such as rounding, but it cannot force the initial step, such as shortening. Only markedness can do so. Tokenized markedness also determines the directionality of movement in a scenario.

In the PC proposal, generalized faithfulness and tokenized markedness constraints are in themselves a combination of markedness and faithfulness constraints. They both access information about the output (like standard markedness) and information about input-output disparity (like standard faithfulness). The question arises: are faithfulness and markedness ever distinct? Where is the boundary between the two types of constraints? PC theory suggests that the boundary between faithfulness and markedness is no longer clear-cut, and constraints infringe on the territory of markedness and faithfulness. The role of such constraints and their properties are further investigated in the following chapters.