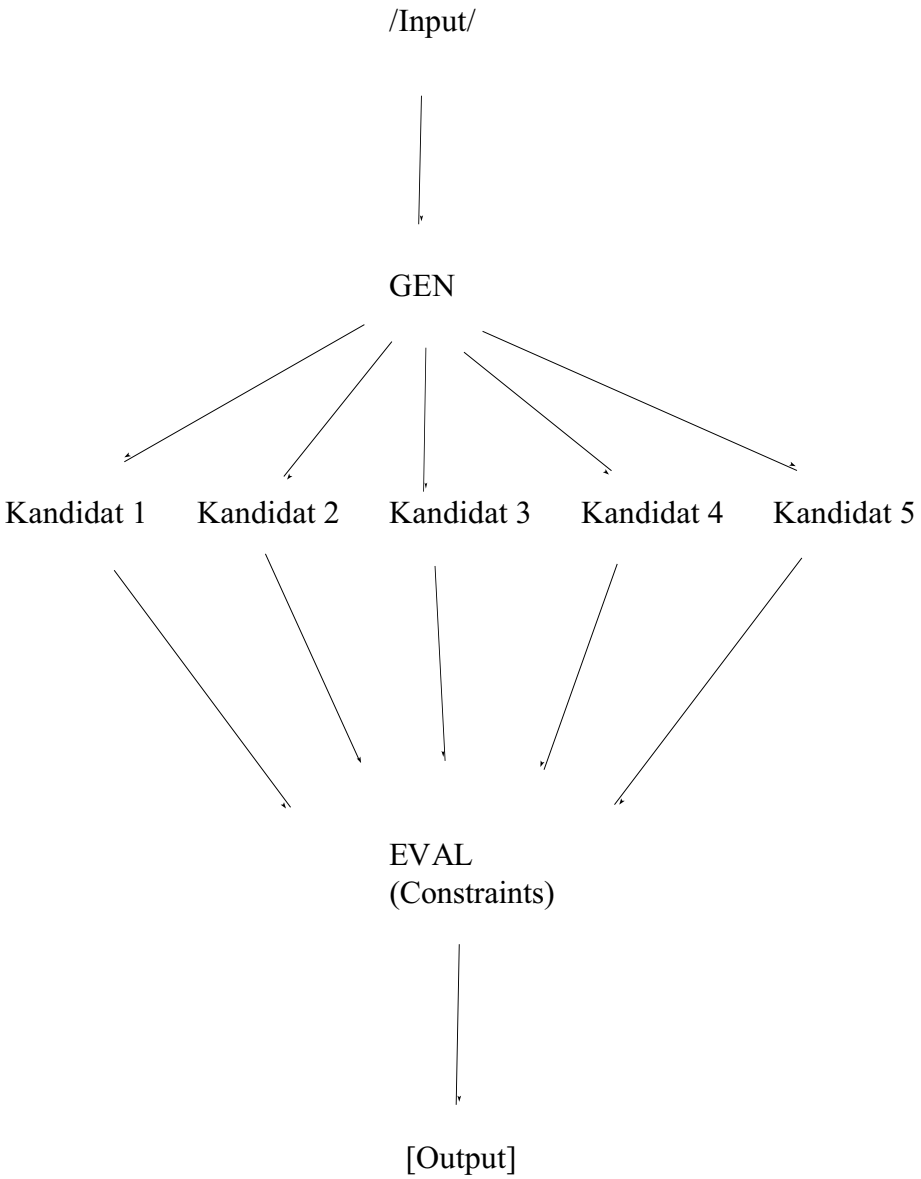
1. Der Aufbau von OT-Grammatiken

1.0 Einleitung

Die Optimalitätstheorie (OT) wurde in den 1990er Jahren entwickelt und wurde in ihrem Gesamtumfang erstmals von Prince/Smolensky (1993) postuliert. Wie in den generativen Ansätzen wird hier eine zugrunde liegende Repräsentation (*underlying representation*, UR) als Input betrachtet. Die Oberflächenrepräsentation (*surface representation*, SR) wird allerdings nicht durch aufeinander folgende Regeln generiert, sondern in der OT liegen zwischen Input und Output, der Oberflächenrepräsentation, die Komponenten GEN und EVAL. Die Komponente GEN stellt eine auf der Grundlage des Inputs generierte, große Anzahl von möglichen Outputs zur Verfügung und erzeugt eine prinzipiell unendliche Menge von Kandidaten. Die Evaluierungskomponente EVAL, die im Zentrum der Theorie steht, wählt den jeweils richtigen Kandidaten für den Output als „optimalen“ aus:

(1)



Es erfolgt keine Ableitung stufenweise wie in den regelbasierten Ansätzen, sondern die OT ordnet einem Input ohne derivationelle Zwischenschritte oder -ebenen eine Oberflächenrepräsentation als Output direkt zu. Bei einem generativ erzeugten Produkt kann es sich dagegen auch um ein Zwischenergebnis handeln, das stets nach einer weiteren Verarbeitung verlangt, solange es keine abgeschlossene Oberflächenrepräsentation darstellt. Zur Veranschaulichung der Abläufe bei der Bewertung von Kandidaten werden in der OT sog. Tableaus (Tabellen) benutzt. Hier werden stets nur diejenigen Kandidaten aufgeführt, die in eine engere Auswahl kommen oder die sich auf markante Weise vom optimalen Kandidaten unterscheiden. Dieser

kann selegiert werden, nachdem sämtliche Kandidaten die Constrainthierarchie durchlaufen haben. Die Constraintabfolge eines sprachlichen Systems enthält in einer sprachspezifischen hierarchischen Anordnung die Constraints (Beschränkungen) der universalen Constraintmenge CON. Sie stellt eine zentrale Komponente der einzelsprachlichen Grammatik dar. Hier passiert jeder Kandidat den am höchsten gerankten Constraint zuerst, dann den nächstniedrigeren usw. Es ist dabei ausgeschlossen, dass ein höher gerankter Constraint von einem niedrigeren überschrieben wird; die Hierarchie der Constraintabfolge wird streng eingehalten. Constraints stehen in Konkurrenz zueinander und können grundsätzlich auch verletzt werden, ohne dass ein Kandidat gänzlich von der Bewertung ausscheidet. Wenn ein von GEN erzeugter Kandidat einen Constraint verletzt, bedeutet dies zunächst lediglich, dass es sich um eine in Bezug auf den spezifischen Constraint markierte Struktur handelt und nicht, dass der Kandidat in jedem Fall im weiteren Verlauf nicht mehr berücksichtigt wird. Gleichzeitig bedeutet dies, dass ein Constraint in einem sprachlichen System umso stärker ausgeprägt ist, je schwächer er verletzt wird. Es sind in keiner Einzelsprache sämtliche Constraints aus der Universalgrammatik aktiv, diese sind aber in jeder Grammatik enthalten. So kann es bei bestimmten Constraints der Fall sein, dass sie in einer Sprache nicht oder nur schwach ausgeprägt sind. Diese Constraints sind am niedrigsten gerankt und sind so in der Hierarchie angesiedelt, dass ihre Verletzung fast keinen oder gar keinen Ausschlag für die Bewertung und bei der Auswahl des optimalen Kandidaten hat. Wird ein Constraint von einem Kandidaten nicht verletzt, so ändert dies nichts an der Aktivität der ihm vorhergehenden und der nachfolgenden Constraints; diese bleiben aktiv. Der höher gerankte Constraint dominiert in diesem Sinn direkt den nachfolgenden. Ist die Verletzung eines Constraints nicht zu umgehen, so wird stets die geringstmögliche Verletzung des Constraints mit dem niedrigsten Rang vorgenommen. Eine Verletzung von niedriger gerankten Constraints wird somit eher in Kauf genommen als die Verletzung eines höher gerankten Constraints. Die Verletzung von Constraints ist infolgedessen stets minimal. Dadurch, dass die Constraints "soft" (Gilbers/de Hoop, 1998, 2) sind, dass sie also verletzt werden können und sollen, ist es auch möglich, sie allgemeiner als die Regeln generativer Ansätze zu formulieren. Hier liegen zwei wichtige Unterschiede zu den regelbasierten Ansätzen: Dort handelt es sich 1. um sprachspezifische Regeln, die 2. nicht verletzt werden dürfen.

Durchläuft ein Kandidat einen Constraint, so stellt dies stets einen unabhängigen Vorgang dar, der nicht in Relation zu Begegnungen mit vorhergehenden oder nachfolgenden Constraints steht. Der Kandidat wird stets streng in Bezug auf einen einzigen Constraint bewertet, sein Gesamtprofil oder die Ergebnisse anderer Constraintbegegnungen sind hier unerheblich (*strict domination*). Verletzen zwei konkurrierende Kandidaten denselben Constraint und stehen keine weiteren Kandidaten zur Verfügung, so ist für den weiteren Verlauf entscheidend, wie oft die beiden Kandidaten denselben Constraint verletzen (vgl. (2, 3)). Ist die Anzahl der Verletzungen gleich, muss über den weiteren Verlauf in den nachfolgenden Constraintbegegnungen entschieden werden. Je größer die Anzahl der verletzten Constraints ist und je öfter ein Kandidat ein und denselben Constraint verletzt, desto mehr *violation marks* (*) erhält er und desto unwahrscheinlicher ist es, dass er durch EVAL als der "optimale" ausgewählt wird. Verletzt aber ein Kandidat einen einzelnen Constraint öfter als ein anderer Kandidat, so scheidet er aus und erhält die Kennzeichnung "!" in dem Feld desjenigen Constraints, in dem die entsprechende Verletzung vorgenommen wird. Diese wird auch "fatal" genannt. Derjenige Kandidat, der die wenigsten Verletzungen von Constraints möglichst niedrigen Ranges aufweist, ist der optimale und erhält im Tableau das Zeichen "☞". Schematisch lässt sich ein OT-Tableau, das zur besseren Übersicht zunächst lediglich zwei Kandidaten und zwei Constraints beinhaltet, folgendermaßen darstellen:

(2)

/Input/	CONSTRAINT 1	CONSTRAINT 2
☞ Kandidat A		*
Kandidat B	*!	

Kandidat B verletzt das Constraintschema auf fatale Weise und scheidet aus, weil er den ranghöheren Constraint verletzt. Dadurch kann bereits, nachdem die Kandidaten CONSTRAINT 1 durchlaufen haben, Kandidat A als der optimale ausgewählt werden. Alle nachfolgenden, rangniedrigeren Constraints kommen für die Untersuchung nicht mehr in Betracht. Die in (3) dargestellten Verstöße der Kandidaten A und B gegen CONSTRAINT 2 und CONSTRAINT 3 wiegen gleich, obwohl CONSTRAINT 2 weiter links als CONSTRAINT 3 steht. Die Trennlinie zwischen den Spalten der beiden Constraints deutet hier die Aufhebung der Rangordnung an.

Die fatale Verletzung erfolgt erst im letzten betrachteten Constraint:

(3)

/Input/	CONSTRAINT 1	CONSTRAINT 2	CONSTRAINT 3	CONSTRAINT 4
Kandidat A		*		***!
☞ Kandidat B			*	**

Je mehr Constraints ein Kandidat verletzt und je öfter er sie verletzt, als desto markierter gilt er. Ist ein Kandidat stärker markiert als ein anderer, scheidet er aus. Die OT strebt somit nach möglichst unmarkierten phonologischen Strukturen. Dies führt zu einem Phänomen, das als "Emergence of the unmarked" bezeichnet wird (McCarthy/Prince, 1994). In letzter Konsequenz müsste dies bedeuten, dass menschliche Sprache in ihrer Phonologie auf einen Schwa-Laut [ə], den unmarkiertesten aller Sprachlaute, zustrebt. Unmarkiertheit ist hier in dem Sinne zu verstehen, dass Schwa für kein Ortsmerkmal und lediglich für das laryngale Merkmal [±stimmhaft] positiv spezifiziert werden kann. Dass menschliche Sprache nicht auf den Zentralvokal Schwa zustrebt, liegt darin begründet, dass Markiertheitsconstraints (*markedness constraints*), die unmarkierte Oberflächenstrukturen verlangen, in einem antagonistischen Verhältnis zu Treueconstraints (*faithfulness constraints*) stehen. Markiertheitsconstraints postulieren jeweils Anforderungen an den Output, während Treueconstraints dafür sorgen, dass die phonologischen Outputformen in einem Treueverhältnis zum jeweiligen Input stehen. Durch Markiertheitsconstraints, die markierten Strukturen entgegenwirken und sie ggf. tilgen, können die im Input vorhandenen Eigenschaften entweder verloren gehen, weil sie im sprachlichen System als markiert anzusehen sind, oder entsprechende Treueconstraints sorgen dafür, dass sie erhalten bleiben. Markiertheitsconstraints beziehen sich ausschließlich auf den Output, während Treueconstraints sich gleichermaßen auf Input und Output beziehen. Im Zusammenspiel dieser Constraintfamilien können Markiertheits- von Treueconstraints dominiert werden und umgekehrt. Ein höher gerankter Treueconstraint bewirkt, dass ein Markiertheitsconstraint, der sich auf ein bestimmtes Merkmal bezieht und dieses seiner Funktion nach tilgt, nicht wirksam wird. Dadurch, dass Treueconstraints einen Bezug zum Input herstellen, wird durch sie eine Bewertung vorgenommen, die an anderer Stelle, in EVAL, fortgesetzt wird (Kager, 1999, 10). Auf diese Weise wird die Generierung von Kandidaten, die später in EVAL ausscheiden würden,

vermieden. Die OT kennt jedoch keine Constraints, die ausschließlich im Input wirksam werden. Der Input kann demzufolge vielfältige Eigenschaften haben (*Richness of the base*). Weiterhin können "Richness of the Base" zufolge auch mehrere Inputs für einen einzigen Output in Frage kommen. Durch die Tableaus ist es in der OT möglich, anhand eines Outputs den zugehörigen Input eindeutig zu bestimmen. Dagegen ist ein derivationeller Ableitungsprozess von der Oberflächenrepräsentation als Endergebnis aus gesehen gewöhnlich nicht rekonstruierbar. Die phonologischen Regeln interagieren somit opak, denn ein phonologischer Prozess verdeckt das Ergebnis des vorhergehenden (Féry, 2000, 183). Auch in der Komponente GEN werden keine Constraints aktiv. Die einzige Beschränkung ist hier, dass die erzeugten Kandidaten aus universalsprachlich unmarkierten Elementen bestehen. Weitere Prinzipien, die GEN zugrunde liegen, sind *Freedom of Analysis*, *Containment* und *Consistency of Exponence*. *Freedom of Analysis* bezeichnet die Fähigkeit von GEN, eine Struktur aus dem Input in prosodischer und in segmentaler Hinsicht zu modifizieren. Ein wichtiges Merkmal des Inputs ist, dass in ihm alle Segmente und sämtliche anderen Eigenschaften enthalten sind und vor allem hier und auch in den Folgeschritten nicht mehr entfernt werden können. Die vollständige Information des Inputs muss daher in sämtlichen erzeugten Kandidaten enthalten sein. In GEN kann somit keine Tilgung mehr erfolgen (*Containment*). Durch GEN können dem Input jedoch Informationseinheiten hinzugefügt werden und so im Kandidaten in Erscheinung treten. Das Prinzip *Consistency of Exponence* geht über diese Anforderungen hinaus, indem es sich auf Morpheme bezieht und besagt, dass keine Veränderungen am Erscheinungsbild eines lexikalisch vollständig spezifizierten Morphems mehr erlaubt sind. Insbesondere sind epenthetische Elemente, die morphologische Alternationen mit sich bringen, ausgeschlossen (McCarthy/Prince, 1993, 89).

1.1 Generalized Alignment

Alignment-Constraints werden nach dem Schema ALIGN formuliert, das für sämtliche *Alignment-Constraints* gültig ist und das daher *Generalized Alignment* genannt wird. *Alignment-Constraints* verlangen zumeist eine kongruente Positionierung der Ränder morphologischer und prosodischer Einheiten. Die Konstituenten, auf die sich *Alignment-Constraints* beziehen, haben stets linke und rechte Ränder, die jeweils miteinander in Übereinstimmung gebracht werden

können. In der Syntax des Constraints kommt auch zum Ausdruck, auf welche Seite dieser Konstituenten sich der Constraint bezieht: ALIGN (Stamm, R, σ, R) bezieht sich auf die rechten Ränder der Silbe und des Stamms, die kongruieren sollen. Aber auch die Position der Konstituenten innerhalb der Syntax eines Constraints ist nicht arbiträr. Sie wird daher auch als *asymmetrisch* bezeichnet. Der ersten Konstituente wird der Alloperator ∀ und der zweiten der Existenzoperator ∃ zugewiesen. Die folgenden *Alignment-Constraints* werden demzufolge unterschiedlich paraphrasiert:

(4) ALIGN (Stamm, R, σ, R)
"Für alle Stämme gibt es mindestens ein Silbe, deren rechter Rand mit dem rechten Rand des Stammes kongruiert."

(5) ALIGN (σ, R, Stamm, R)
"Für jede Silbe gibt es mindestens einen Stamm, dessen rechter Rand mit dem rechten Rand der Silbe kongruiert."

Aus dieser divergierenden Syntax resultieren z.T. gänzlich unterschiedliche Anforderungen an die Kandidaten, die den Constraints Genüge tun müssen. Ein Stamm $[\sigma\sigma]_{\text{Stamm}}$, der aus zwei Silben besteht, erfüllt beispielsweise (4), weil eine Silbe stets mit dem rechten Rand kongruiert. Dieselbe Form erfüllt aber nicht (5), da dieser Constraint von jeder Silbe verlangt, am rechten Rand eines Stammes zu stehen. Dieses Kriterium wird dann von der initialen Silbe nicht erfüllt. Theoretisch müssen nicht immer gleiche Ränder, also zwei linke oder zwei rechte, kongruieren. So besteht die Möglichkeit, einen linken und einen rechten Rand in Übereinstimmung zu bringen und eine Verknüpfung von morphologischen oder phonologischen Konstituenten vorzunehmen. Dies verlangt etwa der Constraint ALIGN-SFX (McCarthy/Prince, 1993, 138), durch den eine prosodische Kategorie und eine morphologische Kategorie zueinander in Beziehung gesetzt werden. Wie können Verstöße gegen *Alignment-Constraints* gewertet werden? In Frage kommen absolute vs. graduelle Verstöße. Ein absoluter Verstoß impliziert eine Dichotomie, d.h. ein Kandidat erfüllt entweder einen Constraint oder er verstößt gegen ihn. Ein Verstoß wird im Tableau durch "*" im entsprechenden Feld gekennzeichnet. Die Anzahl der Sterne bezeichnet dabei die Anzahl der Verstöße, nicht etwa deren Intensität oder die Intensität eines einzelnen Verstoßes. Für die graduellen Constraints (*gradient constraint*) wird dagegen auch die Intensität ihrer Verletzung verzeichnet. Hier wird die Wertung nach der

Schwere, mit der ein Kandidat gegen einen Constraint verstößt, vorgenommen. Die Bewertung von *Alignment-Constraints* erfolgt in diesem Sinne stets skalar. So wird mit der Anzahl der *violation marks* (*) die tatsächliche Entfernung der Ränder angegeben, die nach Maßgabe der Constraints in Kongruenz zueinander stehen sollten. Für eine graduelle Constraintbewertung muss jeweils eine Einheit bestimmt werden, anhand derer die Bewertung vorgenommen wird. Diese erfolgt in den meisten Fällen in Segmenten. Beziehen sich aber *Alignment-Constraints* auf Füße, so wird die Angabe der Entfernung von der Grenze des morphologischen Gegenstücks in Silben angegeben (Kager, 1999, 121).

1.2 Korrespondenztheorie

Die *Korrespondenztheorie* (*Correspondence Theory*) stellt ein System im Rahmen der OT dar, mit dem Strukturen, etwa aus dem Input und dem Output, zueinander in Beziehung gesetzt werden können. Die Treueconstraints PARSE und FILL der allgemeinen OT werden hier durch die Constraints MAX und DEP ersetzt. Diese können auch Beziehungen zwischen verwandten phonologischen Strukturen (d.h. Outputs) zum Ausdruck bringen. Durch *Korrespondenz-Constraints* wird beschrieben, wie die Beziehung zweier Strukturen im Sinne von Korrespondenz gestaltet sein muss. Die Korrespondenzbeziehung wird durch das Symbol $\Re$ (für *Relation*) bezeichnet. In jedem Fall kann eine Korrespondenzbeziehung zweier Ketten wie folgt definiert werden:

(6) Bei zwei Ketten K_1 und K_2 ist Korrespondenz eine Beziehung $\Re$ zwischen Elementen aus diesen Ketten. Die Elemente $\alpha \in K_1$ und $\beta \in K_2$ gelten als korrespondierend, wenn $\alpha \Re \beta$.

Bei Elementen, die in einer Korrespondenzbeziehung zueinander stehen, kann es sich um Segmente, morphologische und prosodische Einheiten handeln. *Containment* ist nun kein notwendiges Prinzip mehr, weil die zugrunde liegenden Formen direkt in Beziehung zu den Oberflächenformen gesetzt werden und kein Zwischenschritt über GEN mehr erforderlich ist. *Korrespondenz-Constraints* sind etwa MAX-IO, DEP-IO, IDENT(F), CONTIGUITY, ANCHORING, LINEARITY, UNIFORMITY und INTEGRITY. Die Korrespondenzbeziehung bezieht sich entweder auf eine *Domäne* (im Input) oder einen *Bereich* (im Output). Die beiden Begriffe werden wie folgt definiert (Kager, 1999, 248f):

(7) Domäne($\mathfrak{R}$): für eine Beziehung $\mathfrak{R} \subset A \times B$, $x \in$ Domäne($\mathfrak{R}$), wenn $x \in A$ und $\exists y \in B$, sodass $x \mathfrak{R} y$

(8) Bereich($\mathfrak{R}$): $y \in$ Bereich($\mathfrak{R}$), wenn $y \in B$ und $\exists x \in A$, sodass $x \mathfrak{R} y$

In den o.g. Definitionen ist eine Kette K_1 als Menge von Elementen kodiert. Die Korrespondenzbeziehung $\mathfrak{R}$ von Elementen der Ketten K_1 und K_2 stellt eine Teilmenge oder jegliche Teilmenge von $K_1 \times K_2$ dar. Somit befindet sich *x* als Element von A genau dann in der Domäne von $\mathfrak{R}$, wenn *y* als ein Element von B vorhanden ist, sodass *x* und *y* durch $\mathfrak{R}$ zueinander in Beziehung gesetzt werden. Gleichermaßen ist *y* als Element von B im Bereich von $\mathfrak{R}$, wenn es ein *x* als Element von A gibt, so dass *x* und *y* durch $\mathfrak{R}$ zueinander in Beziehung gesetzt werden. So hat etwa die erste Gruppe von *Korrespondenz-Constraints*, MAXIMALITY, als ihre *Domäne* die Menge von Elementen aus K_1, dem Input. Der *Bereich* der MAXIMALITY-Constraints ist die Menge von Elementen, die in einer Korrespondenzbeziehung zur Domäne steht, also einer Teilmenge von K_2, dem Output. Es folgen Darstellungen der *Korrespondenz-Constraints* im Einzelnen. Hier wird in den ersten beiden Beispielen (9) und (10) deutlich, wie die *Generalized Correspondence Constraints* MAX und DEP für eine Anwendung spezifiziert werden:

(9) MAX-IO:

Jedes Element der Kette K_1 entspricht (korrespondiert mit) einem Element der Kette K_2., wobei gilt: Domäne($\mathfrak{R}$)=K_1

Es handelt sich um den Constraint MAXIMALITY, der auf eine Korrespondenz zwischen Input und Output bezogen wurde. Er verlangt hier eine maximale Übereinstimmung von Input und Output. Verstößt ein Kandidat gegen ihn, so kommt dies phonologischer Tilgung gleich (McCarthy/Prince, 1995, 260). Bei Reduplikationsvorgängen kommt auch eine Basis und damit ein Output als Domäne in Frage.

(10) DEP-IO:

Jedes Element der Kette K_2 entspricht (korrespondiert mit) einem Element der Kette K_1., wobei gilt: Bereich($\mathfrak{R}$)=K_2

Der Constraint verlangt aus umgekehrter Blickrichtung wie MAXIMALITY, dass jedes Segment des Outputs eine Entsprechung im Input hat. Wird dieser Constraint verletzt, befinden sich

Elemente im Output, die im Input nicht konstituiert wurden, d.h. es liegt ein Epenthesevorgang vor.

(11) IDENT(F):

Korrespondierende Segmente haben für das Merkmal F (*feature*) gleiche Werte, wobei gilt: Wenn x$\Re$y und für x [γF] gilt, dann hat y ebenfalls den Wert [γF].

Dem Constraint IDENT(F) gemäß müssen im Input und Output die Merkmale (*features*) der betrachteten Elemente übereinstimmen. Der Constraint ist als Treueconstraint aus der allgemeinen OT geläufig und wurde in die Korrespondenztheorie integriert.

(12) INTEGRITY:

Kein Element aus K_1 hat mehrere entsprechende Elemente in K_2, wobei für den Fall $x \in K_1$ und $w, z \in K_2$ gilt: wenn x$\Re$w und x$\Re$z, dann w=z.

Ein weiterer wichtiger *Korrespondenz-Constraint* lautet INTEGRITY, der gebietet, dass ein Element der Kette des Inputs nicht mehreren Elementen der zugehörigen Kette des Outputs entsprechen darf (McCarthy/Prince, 1995, 372).

(13) UNIFORMITY:

Kein Element der Kette K_2 hat mehrere zugehörige Elemente in K_1 ("Keine Koaleszenz"), wobei für den Fall $x,y \in K_1$ und $z \in K_2$ gilt: wenn x$\Re$z und y$\Re$z, dann x=y.

Der Constraint richtet sich gegen segmentale Koaleszenzvorgänge und besagt, dass kein Element aus der Kette des Outputs mehrere Entsprechungen im Input haben darf (McCarthy/Prince, 1995, 371).

(14) CONTIGUITY:

Der Constraint CONTIGUITY fordert, dass ununterbrochene Ketten des Inputs im Output erhalten bleiben. Es darf innerhalb der Kette K_1 kein Segment übersprungen oder eingefügt werden. Dieser Constraint bezieht sich somit auf Epenthese- und Tilgungsvorgänge gleichermaßen (McCarthy/Prince, 1995, 260f, 371). Genauer kann man CONTIGUITY aufteilen und I-CONTIGUI-

TY bezogen auf den Input von O-CONTIGUITY bezogen auf den Output abgrenzen (ebd.). I-CONTIGUITY verlangt, dass in der Kette K_1 des Inputs beim Parsen kein Element übersprungen werden darf. Folglich darf O-CONTIGUITY gemäß kein Element in die Kette K_2 im Output eingefügt werden.

(14a) I-CONTIG
Dasjenige Element von K_1, das in Korrespondenz zu einem anderen Element steht, stellt eine ununterbrochene Kette dar ("nichts überspringen"), sodass die Domäne($\Re$) eine einzige, ununterbrochene Kette in K_1 repräsentiert.

(14b) O-CONTIG
Dasjenige Element von K_2, das in einer Korrespondenzbeziehung zu einem anderen Element steht, stellt eine ununterbrochene Kette dar ("nichts einfügen"), sodass der Bereich($\Re$) eine einzige, ununterbrochene Kette in K_2 repräsentiert.

Steht somit *xyz* im Input und *xz* im Output, so verletzt der Output den Constraint I-CONTIG, weil der Bereich von $\Re$ mit {x,z} angegeben wird und *xz* keine ununterbrochene Kette des Inputs repräsentiert. O-CONTIG verbietet eine Epenthese im Output. Wird mit dem Input *xz* der Output *xyz* erzeugt, so ist O-CONTIG hier durch das epenthetische Element *y* verletzt. Wird analog dem Input *xy* der Output *xyz* zugeordnet, so wird eine unmarkierte Struktur in Bezug auf O-CONTIG erzeugt.

(15) LINEARITY:

K_1 stimmt mit der Struktur von K_2 überein und umgekehrt ("keine Metathese"), wobei für den Fall $x, y \in K_1$ und x' und $y' \in K_2$ gilt: wenn $x \Re x'$ und $y \Re y'$, dann $x < y$, wobei $\neg (y' < x')$.

Es handelt sich um einen gegen Metathesevorgänge gerichteten Constraint, der dafür sorgt, dass die Segmentreihenfolge, die durch den Input vorgegeben wird, im Output die gleiche bleibt. Gleichwohl bleiben Metathesevorgänge möglich (McCarthy/Prince, 1995, 261).

(16) ANCHORING:

{RIGHT, LEFT}- ANCHOR(K_1,K_2)

Jedes Element an dem bezeichneten (rechten/linken) Rand der Kette K_1 hat eine Entsprechung an dem bezeichneten (rechten/linken) Rand der Kette K_2,
wobei unter der Bedingung, dass *Rand* (X, {L, R}) = das Element am Rand L, R von X ist, gilt:
für RIGHT-ANCHOR: ist x = Rand(K_1, R) und y = Rand(K_2, R) dann x$\Re$y,
für LEFT-ANCHOR: ist x = Rand(K_1, L) und y = Rand(K_2, L) dann x$\Re$y.

Der Constraint verlangt für Input-Output-Beziehungen eine kongruente Anordnung von morphologischen und prosodischen Konstituenten und kann innerhalb der Korrespondenztheorie die von *Alignment-Constraints* erfassten Vorgänge modellieren (McCarthy/Prince, 1995, 261). Der Constraint ist vor allem für Präfigierungs- und Suffigierungsprozesse maßgeblich.