Quadratic Alignment Constraints and Finite State Optimality Theory

Tamás Bíró
Department of Humanities Computing
Rijksuniversiteit Groningen
birot@let.rug.nl

Abstract

Previous approaches to Finite State Optimality Theory have supposed that one can build transducers modelling the behaviour of the constraints, *e.g.* that would assign the correct number of violation marks to the candidates. A constraint is called linear if there is a linear function of the input string's length that is an upper bound on the number of violation marks assigned. Quadratic constraints can assign a number of marks quadratic in the input's length. We shall prove that only linear constraints can be realized as a finite state transducer. Some widely used alignment constraints, *e.g.* for stress assignment, are not linear. Interestingly, these constraints have also been criticized in recent phonological literature.

1 Introduction

Optimality Theory (OT) has been a leading paradigm in linguistics, especially in phonology, since its appearance (Prince and Smolensky, 1993). Computational aspects of OT have been investigated already since its earliest appearance.

According to the basic suppositions of OT, the grammar is composed of two parts: the *Gen* module generates a (possibly infinite) set of *candidates* out of the given underlying representation, while the *Eval* module determines the optimal element of this set. The optimal element(s) will become the grammatical form(s) (the surface representation, the output of the production process). There is a universal set of *constraints*, each of them assigning a given number of violation marks to each of the candidates. For each constraint these violation marks define a strict partial order called *harmonic ranking* on the set of the candidates. For each language there is a (fully) ranked hierarchy (*i.e.* a sequence of application) of these constraints, determining which candidate will be the optimal one chosen by Eval. Within the latter, the highest ranked constraint will filter out the candidates in favour of alternative competitors that are assigned fewer violation marks (being "more harmonic" according to harmonic ranking). Then the second highest ranked constraint will filter out further elements of the remaining set of candidates, using the same method, etc.

In the last years research has been carried out dealing with the question whether Optimality Theory can be implemented using finite state technology (Frank and Satta, 1998; Karttunen, 1998; Gerdemann and v. Noord, 2000; Jäger, 2002). Based on the claim that phonology can be best approximated in fact as a regular (rational) relation between the underlying representation and the surface form (Johnson, 1972; Koskenniemi, 1983; Kaplan and Kay, 1994; Bird and Ellison, 1994), an OT model for phonology should also be realizable with finite state transducers (FSTs), supposing that the model is adequate and not too powerful for phonology.

The idea of Finite State Optimality Theory is to regard the grammar as the composition of finite

state transducers. The first one represents Gen, and produces the set of the candidates when inputting an underlying form. The constraints constituting Eval act as filters, outputting the harmonical candidate(s) of their input set. They are composed by the "optimality operator" (`oo`) in a serial way, following the actual hierarchy:

```
gen oo con1 oo con2 oo ...  oo conN
```

The feasibility of Finite State Optimality Theory consists of three components. The first and least explored one is asking which linguistic models use a Gen that can be formulated as a (non-deterministic) transducer. Previous work (Karttunen, 1998; Gerdemann and v. Noord, 2000) has used the syllabification example – the classical example since (Prince and Smolensky, 1993) – and they have shown the Gen of this paradigm to be a regular relation. Ongoing work shows that a finite transducer can be written that realizes the Gen of the OT model for metrical structure and stress. It would be a challenging task to investigate what criteria a linguistic model should meet for its Gen to be a regular relation (see *e.g.* reduplicative morphologies).

The second question, the most elaborated so far, asks if it is possible to build a model (an optimality operator), supposing one has the required transducers for Gen, as well as some sort of transducer for each of the constraints. Frank and Satta (1998) prove that this is possible by using *lenient composition*, if constraints assign maximally one violation mark to each candidate. If we build a series of n filters for each constraint, gradually filtering out those candidates having at least 1, then at least 2, etc. violations (supposing there are better candidates, otherwise letting all pass), we can realize an OT-system for the case when there is an upper bound n on the number of violation marks assigned to a candidate ("counting approach"). Karttunen (1998) implements this idea for the syllabification paradigm.

The "matching approach" proposed by Gerdemann and van Noord (2000) does not need an upper bound on the number of violations. It also uses transducers assigning violation marks for each constraint, but the key idea is to create a set including the non-optimal candidates by adding extra violation marks. The output should match the complement of this set (the latter may also include strings not being a candidate). Because of the construction, exactness is not always guaranteed, and sometimes only an approximation can be achieved, although it performs better then the "counting approach".

The recent proposal by Jäger (2002) generalizes the results of Gerdemann and van Noord (2000), and proves that an OT system can be realized under certain conditions: the OT model uses only rational output markedness constraints and optimality is global. The point that concerns us the most is that constraints must be rational. Informally speaking, a constraint is said to be "rational" if there exists a rational (regular) relation S such that for any two candidates x and y, if these can be generated from the same underlying representation then $< x, y > \in S$ iff x is more harmonic than y. In other words, for any input string, S creates the set of worse candidates originating from the same underlying representation, as well as possibly some non competitor strings. In this case, a simple filter can be built from S, using *generalized lenient composition.*

The third question concerning the feasibility of Finite State Optimality Theory is what constraints can be modelled as a finite transducer following the needs of the approach used: assigning violation marks in the case of (Karttunen, 1998) and (Gerdemann and v. Noord, 2000); or mapping a candidate to its less harmonic competitors in the case of (Jäger, 2002).

Only "output markedness" constraints have been considered so far, *i.e.* violations depend only on the form of the candidate, an not on the underlying representation they are derived from. Furthermore, violations should be assigned using standard string manipulation techniques, and some sort of locality is probably also required.

In this paper we shall prove that no violation mark assigning transducer can be built for the type of constraints that have no linear bound on the number of violation marks assigned, in function of the input string's length (non-linear constraints). The examples of this type of constraint come from the bigger family called the alignment constraints (McCarthy and Prince, 1993), and are widely used

in state-of-the-art phonological-morphological literature. These are gradient constraints in the following sense:[1] they can assign more violation marks to the same undesired substring of the candidate, in function of the degree of the violation's seriousness. Gradience in general, and gradient alignment constraints in particular have recently been heavily criticised by McCarthy (2002) on linguistic grounds, that align nicely with my arguments against their use.

2 Gradient Constraints in OT

As mentioned, in the most common formulation constraints assign violation marks to the candidates. A candidate can be assigned multiple violation marks by one constraint, and linguistic literature on Optimality Theory has had three hypotheses about the nature of multiple violations, implicit in (Prince and Smolensky, 1993). Quoting McCarthy (2002):

- *Locus hypothesis*: A violation mark is assigned for each instance or *locus* of violation within a candidate. When presented with a right candidate, then, any OT constraint can assign multiple violation marks.
- *Gradience hypothesis*: Some constraints, by virtue of their formulation, assess violations gradiently. These constraints can assign multiple violation marks even when there is just a single locus of violation.
- *Homogeneity hypothesis*: Multiple violations of a constraint from either source are added together in evaluating a candidate. No distinction is made between multiple violation marks derived from the Locus hypothesis and those derived from the Gradience hypothesis.

Here, we are following McCarthy's terminology, and in this paper a "gradient constraint" is a constraint that can assign multiple violation-marks to the same substring of the candidate, depending on how much disfavoured the given structure is.

McCarthy makes the distinction between two types of gradient constraints. Some (vertical, collective and scalar gradience) are always limited in extent of violation: they can assign 0, 1, 2,... or n violation marks, depending on how serious the violation is. This means that the decision of assigning violation marks is made locally, and the number of violation marks assigned to the whole candidate is maximally linear in the length of the string.

But what McCarthy calls "horizontal gradience", basically the family of alignment constraints discussed in the next section, is different in nature. They assign violation marks in proportion to some distance within the string. Therefore, as we shall see on some examples, the number of violation marks assigned to one candidate may be quadratic in the length of the candidate.

McCarthy claims that gradience is not inherently a property of Optimality Theory (and therefore the homogeneity hypothesis can also be avoided). Gradient constraints with a limited number of violation marks can be rewritten as a series of non-gradient constraints: $C1$, $C2$, ..., Cn, where Ci assigns one violation mark exactly if the gradient version would assign at least i violations. (One would obviously suppose an inherent universal ranking for these constraints.)

Furthermore, after having discussed the relevant OT literature, McCarthy (2002) brings heavy linguistic arguments against "horizontal gradience" constraints. He proposes a new family of constraints ("quantized alignment") instead, that are not gradient. The number of violation marks assigned by them is upper bounded by the length of the input string.

From the point of view of finite state technology, vertical, collective and scalar gradience constraints do not seem to pose a problem, it would not be too hard to build the respective transducers assigning violation marks. The same applies to McCarthy's quantized alignment constraints. But some of the widely used alignment constraints (*e.g.* for stress assignment), criticized by McCarthy, do not correspond to a regular relation, as we shall prove in this paper.

[1] It should be emphasized that the term "gradient constraint" has been used in Finite State OT literature to refer to any type of constraints that can assign more than one violation mark to a candidate. A constraint like Parse, disfavouring the underparsing of some elements of the candidate, would assign one violation mark to each underparsed unit (*cf.* locus hypothesis, bellow), and therefore has been said to be a gradient constraint. But it is not according to the above definition.

3 Alignment constraints for stress

Typical gradient constraints in OT are the so-called "alignment constraints", used mainly for metrical stress assignment and infixation. Here we shall present the example of metrical stress.

The classical way of analysing metrical stress within an OT framework goes back to the very first years of OT (McCarthy and Prince, 1993), based among others on earlier works of Bruce Hayes (1981). Gen assigns a three-level hierarchical metrical structure to each element of the candidate set. Some syllables are organized into feet, and the prosodic word consists of these feet, as well as the syllables that are not parsed into feet. (Unlike in the syllabification example, an unparsed element is still pronounced.) Each foot has a head syllable, and the prosodic word has exactly one head (main) foot. The head syllables are the ones bearing stress: the head syllable of the main foot bears the primary stress, while the head syllables of the non main feet bear secondary stress:

$$\sigma(\sigma\ \sigma 2)[\sigma 1\ \sigma]\sigma(\sigma 2)$$

Here squared brackets refer to the main foot, parentheses refer to the non-main ones. Numbers show the place of stress, 1 stands for primary stress, and 2 for secondary one.

The output of Gen is the set of all possible parses of the input (the underlying form). That is: all possible distributions of main and non-main feet, including all possible distributions of head syllables within each foot.

Typically, constraints refer to some combinations of the ingredients of this model: foot edges, word edges, stress distribution and syllable types. The most interesting constraint family concerns the place of feet within a prosodic word. These are called alignment constraints (McCarthy and Prince, 1993), and their general definition is:

> *Let Cat1 and Cat2 be two categories, and Edge1 and Edge2 be elements of the $\{L, R\}$ set (standing for "left" and "right"). ALIGN(Cat1, Edge1, Cat2, Edge2) is satisfied iff for each substring belonging to Cat1: its edge Edge1 coincides with the edge Edge2 of some substring belonging to Cat2.*

Sometimes *ALIGN(Cat1, Cat2, Edge)* is used when *Edge1* and *Edge2* are the same.

Widely used imlementations of these constraints are ALIGN(Wd, Ft, L/R), ALIGN(Ft, Wd, L/R) and ALIGN(MFt, Wd, L/R).

The first pair of constraints (called Word-Foot-Left and Word-Foot-Right in (Tesar and Smolensky, 2000)) assigns one violation mark if the left (right) edge of the prosodic word does not align with the left (right) edge of some foot. These constraints can assign maximally one violation mark to each candidate, and do not pose any problem to finite state technology.

The four other alignment constraints (called All-Feet-Left/Right and Main-Left/Right in (Tesar and Smolensky, 2000)) are gradient constraints.

Main-Left assigns as many violation marks as the number of syllables intervening between the left edge of the word and the left edge of the main foot. Main-Right does the same for the relevant right edges. Here the number of syllables in the candidate is an upper-bound for the number of violation marks assigned, because a word has exactly one main foot.

The way to realize Main-Right and Main-Left as a finite state transducer would be to reformulate them, in the form of prohibiting a syllable to intervene between the relevant edge of the main foot and the relevant edge of the word. So we would assign one violation mark to each syllable between the two edges, and thus we could escape the gradient nature of these constraints.

But this is not the case for All-Feet-Left/Right. These are "real" gradient constraints. All-Feet-Left for instance will assign to each foot as many violation marks as the number of syllables intervening between the left edge of the word and the left edge of the foot in question. Therefore in the case of the following candidate:

$$[\sigma\sigma](\sigma\sigma)(\sigma\sigma)$$

the first foot whose left edge aligns with the left edge of the word will not be assigned any violation marks, but two violation marks will be assigned to the second foot, and four to the third one. If all syllables were parsed into a separate foot, the candidate would have been assigned 15 marks.

In general, to a candidate consisting of n syllables, All-Feet-Left and All-Feet-Right can assign maximally $n(n-1)/2$ violation marks, and this happens when a candidate has all its syllables (or all but the one being on the relevant edge) footed into a separate foot. This is why we call these two constraints "quadratic constraints".

4 No transducers assigning violation marks for quadratic constraints

The consequence of this last fact is very serious. Intuitively speaking, the function that assigns violation marks according to the constraints All-Feet-Left / Right requires embedded cycles checking a string, which can be only approximated by finite state techniques.

In order to prove mathematically that these two constraints cannot be formulated by FSTs, first we shall present a lemma, that is in fact a simple consequence of the so-called *pumping lemma.*

Lemma: Let **T** be a functional finite state transducer, *i.e.* for any input string σ it produces at most one output $T(\sigma)$. Then there exists a linear upper bound on the length of the output, *i.e.* there exists a positive integer N such that for any non-empty input string σ for which there exists an output $T(\sigma)$, the following inequality holds:

$$\mid T(\sigma) \mid \leq N \mid \sigma \mid$$

where $\mid \alpha \mid$ denotes the length of the string α. □

The proof of this lemma is to be found at the end of the Appendix, in the form of a corollary.

The next step is to realize that All-Feet-Left and All-Feet-Right can assign a number of violation marks that is quadratic in the length of the input. In fact if the input consists for instance of n syllables, each of them parsed into a separate foot, then the number of violation marks to be assigned to the word is $n(n-1)/2$. Therefore no linear bound can be given (in function of the input's length) to the length of the output of the process assigning violation marks. But this process has been supposed to be functional, mapping the input string onto the string including violation marks as well.

Supposing we had a functional transducer realizing the All-Feet-Left or the All-Feet-Right constraint, according to the above lemma there would be an integer N such that the maximum number of violation marks assigned to a word consisting of n syllables would be $N-1$ times the length of the input (no deletion takes place in violation mark assignment). If we suppose that ξ is the maximum length of a syllable[2] we get the inequality:

$$\frac{n(n-1)}{2} \leq (N-1)n\xi$$

But we have to realize that it is possible to choose n great enough so that this would not hold. If the number of syllables is:

$$n > 1 + 2(N-1)\xi$$

then the inequality following from the lemma will not be satisfied. As there is no theoretical limit on the number of syllables in a word, we have proven that no functional finite transducer exists realizing All-Feet-Left or All-Feet-Right in an exact way.

5 Approximations for assigning violation marks

But this does not mean that no approximation can be given. One can suppose in practice that in real life languages, the number of syllables (or even more the number of feet) in one word is bounded.

Here we are giving an approximation for All-Feet-Right. First we build an FST called `one_foot_right` that assigns a violation mark (`@`) to all end-of-syllable symbols (`eos`) right to the foot just being checked (marked by a `C` character). This can be done by using the context sensitive rewrite operator `replace(Transducer, Left_context, Right_context)`, as presented in (Gerdemann and van Noord, 1999):[3]

[2] Such a supposition cannot be made in general. But since violation mark assignment by these two constraints is independent from the phonemes in the word, one could just delete the phonemic content of the input, without altering the process. In such a case an upper limit can already be given, because a syllable consists maximally of the syllable type specification, the stress type symbol, foot brackets and the symbols delimiting words and syllables.

[3] We are using the formalism of FSA Utilities, as introduced by (van Noord, 1997), (van Noord, 1999), (Gerdemann and van Noord, 1999) or (Gerdemann and v. Noord, 2000). `[]` stands for the empty string, `[A, B]` is the concatenation of `A` and `B`, `~C` stands for the set complement of `C`, `?` matches any character, `*` stands for Kleene-star, and `A:B` is a transducer mapping `A` to `B` and anything else into itself.

```
one_foot_right(@) :=
   replace([]:@,[C, eos, ?  *, eos],[]).
```

A step consists of marking the first unchecked foot (not marked by the "checked foot symbol" `D`) from the beginning of the word (`bow`) by symbol "being checked" (`C`) mentioned above; then running `one_foot_right` and finally marking that foot as already checked (`D`; `fr` stands for a right edge of a foot):

```
one_step(@) :=
   replace([]:C,[bow,(~C)*,fr],~D)
            o one_foot_right(@) o (C:D).
```

If we have a bound n on the number of feet in a word, repeating this process n times would result in a realization of All-Feet-Right.

```
mark_ot_constraint(all_feet_right,@) :=
   one_step(@)1 o one_step(@)2 o ...
            o one_step(@)n o (D:[]) .
```

A similar procedure is possible for All-Feet-Left, as well. It is noteworthy that even a three-step approximation of the All-Feet-Right constraint results in an FST that has 472 states. This "explosion" in the number of the states shows the inherently not finite state-ness of the problem.

6 Further possibilities

In Section 4, we have proved that no *functional* FST can be built that would distribute violation marks according to constraints that are supposed to assign a quadratic number of marks, such as is the case for some gradient alignment constraints.

In fact one could argue that there is no need for those transducers to be functional. Suppose the transducer would output a string with the correct number of violation marks, as well as a number of fake candidates, all of them having more violation marks than the correct one. Since these fake candidates are less harmonic than the correct one, they will be eliminated by the optimality operator.

In the case of quadratic constraints, this would require a transducer whose *shortest* output has no linear bound in function of the input's length. In fact this is also impossible due to the first lemma proved in the Appendix:

Lemma: Let **T** be a finite state transducer. Then there exists a linear upper bound on the length of the shortest output, *i.e.* there exists a positive integer N such that for any non-empty input string σ for which $T(\sigma) \neq \emptyset$ the following inequality holds:

$$\min_{\xi \in T(\sigma)} \mid \xi \mid \leq N \mid \sigma \mid \qquad \square$$

Another possibility is to follow the idea presented in (Jäger, 2002), *i.e.* using the idea of (Gerdemann and v. Noord, 2000) but avoiding references to violation marks. (A constraint can be seen as a strict partial order on the set of candidates, for which every subset has a maximal element (Samek-Lodovici and Prince, 1999).)

Suppose that for some constraint `Con1`, an FST `worse_wrt_con(Con1)` generates the subset of candidates that are less harmonic than the input string with respect to `Con1`. The generalized lenient composition, as optimality operator, following (Jäger, 2002) is then:

```
Input oo Con1 :=
   Input o ~(Input o worse_wrt_con(Con1))
```

The second factor of the composition is an identity transduction on the complement of the non-optimal forms. Notice that `worse_wrt_con(Con)` could generate also some strings that are not within the candidate set of the corresponding underlying form.

Therefore future work should either present such a transducer for quadratic alignment constraints, or should prove that no such FST exists (they are not rational constraints according to (Jäger, 2002)).

In the case of All-Feet-Right for instance, such a transducer should among others: add additional feet; move feet towards the left edge; and delete a foot with a distance of k syllables from the right edge, and simultaneously add feet whose summed distance is more than k. The complexity of this last task hints that no such finite state transducer would exist.

7 Conclusion

In this paper we have presented some techniques that can be used to decide whether a constraint can be formulated as a finite state transducer. We have shown that constraints which assign more than a linear number of violation marks cannot be applied within the framework of violation mark as-

signing finite state models. (The generalized issue presented by Jäger (2002) is still open.)

The "non-linear" constraints we know about are gradient constraints that cannot be redefined as a non-gradient constraint: they will always have to assign more than one violation mark to some loci of the candidates. In the case of other gradient constraints, like the first three types in (McCarthy, 2002) with a bounded number of violation marks per locus, or Main-Foot-Left / Main-Foot-Right, we have seen that it was possible to reformulate them in a non-gradient way. And simultaneously, they can be realized as regular relations.

McCarthy (2002)'s independent arguments against gradient constraints, and his proposed non-gradient (and finite-state friendly) alternatives for them, reassures us that this result does not menace our confidence in the finite-stateness of phonology. Also, one could ask what the psychological adequateness of some quadratic constraints would be. But even if it turned out finally that those constraints cannot be dismissed, we have proposed some approximations that could be used given a practical limit on the length of words.

Appendix. Linearity of FSTs

Lemma: Let **T** be a finite state transducer. Then there exists a linear upper bound on the length of the shortest output, *i.e.* there exists a positive integer N such that for any non-empty input string σ for which $T(\sigma) \neq \emptyset$ the following inequality holds:

$$\min_{\xi \in T(\sigma)} \mid \xi \mid \leq N \mid \sigma \mid$$

where $\mid \alpha \mid$ denotes the length of the string α.

Proof:

Suppose that **T** does not accept the empty string as input, otherwise consider a transducer **T'** accepting the same language, except of the empty string. Such a transducer, with an input alphabet A and an output alphabet B can be seen as a finite state automaton over the alphabet $X \subseteq (A \cup \{\epsilon\}) \times (B \cup \{\epsilon\}) \setminus \{(\epsilon, \epsilon)\}$ (*cf.* (Berstel, 1979), remark after corollary 6.2, on p. 79). A string $(a_1, b_1)(a_2, b_2)...(a_n, b_n)$ accepted by the automaton corresponds to the input-output pair $(a_1 a_2 ... a_n, b_1 b_2 ... b_n)$ of the transducer, with the ϵ-s being simply deleted.

For a string $f = (a_1, b_1)(a_2, b_2)...(a_n, b_n)$ accepted by the automaton, let us call the first projection $f_i = a_1 a_2 ... a_n$ the *left-hand* or *input string* of f, and let the second projection $f_o = b_1 b_2 ... b_n$ be the *right-hand* or *output string* of f.

Now we will make use of a corollary of *Ogden's Iteration Lemma for Regular Languages*, a variation of the *Pumping Lemma* (Corollary 4.7 in (Berstel, 1979), p. 21). This claims that if $L \subset X^*$ is a regular language, and $Y \subset X$, then there is an integer $N \geq 1$ such that for any $f \in L$ and for any factorisation $f = hgh'$ with $\mid g \mid_Y \geq N$,[4] g admits a factorisation $g = aub$ such that (i) $0 < \mid u \mid_Y \leq N$, and (ii) $hau^*bh' \subset L$.

Let $L \subset X^*$ be the language accepted by the FSA corresponding to the finite state transducer **T**, as explained above. And let be $Y = \{\epsilon\} \times B \subset X$, corresponding to insertions during the transduction process. This means that there exists a positive integer N, such that for any $f \in L$ and for any factorisation $f = hgh'$: if $\mid g \mid_Y \geq N$, then g can be factorised such as $g = aub$, with $\mid u \mid_Y > 0$ and $hau^*bh' \subset L$.

Let $f \in L$ be such that its right-hand string f_o after deletion of the ϵs is the shortest output $\hat{\xi}$ corresponding to an acceptable string σ (f_i without the ϵs) with respect to the transducer **T**. (Such a string exists since $T(\sigma)$ is denumerable.) So, using the above corollary, we obtain that there is an integer $N \geq 1$ such that for any factorisation $f = hgh'$ with $\mid g \mid_Y \geq N$, g admits a factorisation $g = aub$ such that $\mid u \mid_Y > 0$ and $f' := habh' \in L$.

In this case $u \notin Y^+$ should hold, otherwise f and f' would correspond to the same input of the transducer, but the output corresponding to f' would be shorter than the output corresponding to f, which contradicts our supposition that f encodes a transducing when the input string is mapped onto its shortest possible output.

Therefore we can conclude that for any factorisation $f = hgh'$, if $\mid g \mid \geq \mid g \mid_Y \geq N$ then $g \notin Y^+$, since its non-empty substring u contains also at least one element of $X \setminus Y$.

In other words: it is not possible to find a con-

[4] $\mid u \mid_Y$ refers to the number of occurrences of elements of Y in the string u.

tinuous substring of more then $N-1$ elements of Y within f. Remembering the way we constructed our automaton from the transducer **T**, and realizing that the set Y refers to insertions during transduction, while $X \setminus Y$ refers to reading a symbol of the input string ((ϵ, ϵ)-transitions have been eliminated), we can conclude that not more than $N-1$ characters are inserted into the output after each element of the input.

Since the input string σ is f_i after deleting the ϵs, and the shortest corresponding output $\hat{\xi}$ is f_o after deleting the ϵs:

$$\min_{\xi \in T(\sigma)} \mid \xi \mid = \mid \hat{\xi} \mid \leq \mid f \mid \leq N \mid \sigma \mid$$

Thus we have proven our lemma.□

If T is a functional transducer, that is for any input string σ it produces at most one output $T(\sigma)$, then we obtain the following[5]

Corollary: Let **T** be a functional finite state transducer. Then there exists a linear upper bound on the length of the output, *i.e.* there exists a positive integer N such that for any input string σ for which there exists an output $T(\sigma)$ the following holds:

$$\mid T(\sigma) \mid \leq N \mid \sigma \mid \qquad \square$$

Acknowledgement

I wish to acknowledge the support of the University of Groningen's program for High-Performance Computing, and the help of J. Nerbonne, G. Bouma, G. van Noord and J. Daciuk.

References

Jean Berstel. 1979. *Transductions and Context-Free Languages*. Teubner, Stuttgart.

Steven Bird and Mark T. Ellison. 1994. One-level phonology: autosegmental representations and rules as finite automata. *Computational Linguistics*, 20(1):55–90.

Robert Frank and Giorgio Satta. 1998. Optimality theory and the generative complexity of constraint violability. *Computational Linguistics*, 24(2):307–315.

Dale Gerdemann and Gertjan v. Noord. 2000. Approximation and exactness in finite state optimality theory. In *Jason Eisner, Lauri Karttunen, Alain Thriault (eds): SIGPHON 2000, Finite State Phonology*.

Dale Gerdemann and Gertjan van Noord. 1999. Transducers from rewrite rules with backreferences. In *Ninth Conference of EACL, Bergen, Norway*.

Bruce Hayes. 1981. *A Metrical Theory of Stress Rules*. Indiana University Linguistic Club, Bloomington.

Gerhard Jäger. 2002. Gradient constraints in finite state ot: The unidirectional and the bidirectional case. *ROA-479*[6]

Douglas C. Johnson. 1972. *Formal Aspects of Phonological Description*. Mouton, The Hague [etc.].

Ronald M. Kaplan and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–378.

Lauri Karttunen. 1998. The proper treatment of optimality theory in computational phonology. In *Finite-state Methods in Natural Language Processing*, pages 1–12. Ankara.

Kimmo Koskenniemi. 1983. Two-level morphology: A general computational model for word-form recognition and production. *Publication No. 11, Department of General Linguistics, University of Helsinki*.

John J. McCarthy and Alan Prince. 1993. Generalized alignment. In *Yearbook of Morphology*, pages 79–153. Dordrecht: Kluwer, also: ROA-7.

John J. McCarthy. 2002. Against gradience. *ROA-510*.

Alan Prince and Paul Smolensky. 1993. Optimality theory, constraint interaction in generative grammar. In *RuCCS-TR-2, ROA Version: 8/2002.*

Vieri Samek-Lodovici and Alan Prince. 1999. Optima. *ROA-363*.

Bruce Tesar and Paul Smolensky. 2000. *Learnability in Optimality Theory*. The MIT Press, Cambridge, MA - London, England.

Gertjan van Noord. 1997. Fsa utilities: A toolbox to manipulate finite-state automata. In *Darrell Raymond, Derick Wood and Sheng Yu (eds.): Automata Implementation, Springer Verlag, Lecture Notes in Computer Science 1260*, pages 87–108.

Gertjan van Noord. 1999. Fsa6 reference manual. The FSA Utilities toolbox is available under Gnu General Public License at `http://www.let.rug.nl/~vannoord/Fsa/`.

[5]The next result could be also obtained directly, using a very similar proof, with the only difference that $u \notin Y^+$ follows from the fact that otherwise the transducer would not be functional.

[6]ROA stands for *Rutgers Optimality Archive* at `http://roa.rutgers.edu/`.