HANDLING EQUIVALENCE CLASSES OF OPTIMALITY-THEORETIC COMPARATIVE TABLEAUX

Abstract. Many Optimality-Theoretic tableaux contain exactly the same information, and equivalence-preserving operations on them have been an object of study for some two decades. This paper shows that several of the operations proposed in the earlier literature together are actually enough to express any possible equivalence-preserving transformation. Moreover, every equivalence class of comparative tableaux (equivalently, of sets of Elementary Ranking Conditions, or ERC sets) has a unique and computable normal form that can be derived using those elementary operations in polynomial time. Any equivalence-preserving operation on comparative tableaux (ERC sets) is thus computable, and normal form tableaux may therefore represent their equivalence classes without loss of generality.

Optimality Theory (OT) is a grammatical formalism based on constraint competition, formulated by defined by [Prince and Smolensky, 1993] (later published as [Prince and Smolensky, 2004]). OT is especially popular in phonology, and is used to some extent in other branches of linguistics. In OT, a set of competing output forms \{Out1, Out2, \ldots\} is generated by machine GEN for the underlying form Inp. Each pair \langle Inp, OutM \rangle is then evaluated against a set of constraints Con. The grammar of a particular language is modeled as an ordering of the universal set of constraints Con which determines the winning input-output pair for each Inp: an input-output pair \alpha = \langle Inp, OutM \rangle wins over another pair \beta = \langle Inp, OutN \rangle in case \alpha incurs less violations than \beta in the most highly ranked constraint where \alpha and \beta differ. The input-output pairs that do not lose to any other pair are declared grammatical.

The OT formalism expresses two important intuitions regarding how languages might function. First, it easily captures conditions of the form “try A; if impossible, try B; if also impossible, resort to C”, which seem to frequently occur in natural language (though, of course, for many seeming “defeasibility phenomena” analyses have been proposed that do not invoke actual defeasibility of constraints or rules). Second, OT allows for elegant modeling of cross-linguistic variation and language change in terms of re-ranking of a universal set of constraints.

The information that a given dataset contributes constrains the possible rankings of constraints. Such information may be represented in the form of a comparative tableau ([Prince, 2000]) or the corresponding set of Elementary Ranking Conditions.

Date: August 14, 2013.

Key words and phrases. Optimality Theory, ERC set, tableau equivalence, normal form, equivalence-preserving transformations.
or ERC set ([Prince, 2002]). In this paper, I present an incremental step completing
the development of a full theory of equivalence classes of comparative OT tableaux,
or, equivalently, ERC sets.

Earlier work, especially [Hayes, 1997], [Prince, 2000], [Prince, 2002], [Brasoveanu and Prince, 2005]
(later published as [Brasoveanu and Prince, 2011]), and [Prince, 2006], has estab-
lished a number of results concerning how one may transform the information in an
OT tableau without loss. What has not yet been done in this line of research is to
study the limits of operations that preserve equivalence. For example, the following
simple question has not been answered: given two arbitrary comparative tableaux
or ERC sets, can we determine whether they contain identical information?1

The present paper fills this gap: I show that any (finite) comparative tableau
may be (computably, and actually quite efficiently) transformed into a normal form,
which is unique for the whole equivalence class. Moreover, this transformation is
possible by applying a sequence of a small set of elementary operations already having
been introduced in the literature. We thus get a handle on equivalence classes of
tableaux/ERC sets: each equivalence class contains exactly one normal form tableau
that may serve as its representative. A (computable) test for equivalence of (finite)
arbitrary tableaux involves normalizing both of them and comparing the resulting
normal form tableaux: the original tableaux are equivalent if and only if their normal
forms are identical. Thanks to the normal form theorem proved in the present paper,
the space of all possible equivalence-preserving operations may be enumerated.

1. Introduction

As a concrete example of how OT works, consider the pattern of final obstruent
devoicing in Dutch.2 Underlyingly, Dutch morphemes may have both voiced and
voiceless obstruents: the morpheme for ‘bed’ is /bed/, surfacing faithfully in [bEd-
@n] ‘beds’, while the morpheme for ‘dab’ is /bet/, surfacing faithfully in [bet-on]
‘(we) dab’. But when the final obstruent of either morpheme closes the syllable, it is
realized on the surface by the same voiceless [t]: both ‘bed’ /bed/ and ‘(I) dab’ /bet/
surface as [bEt]. The following OT tableau demonstrates the violation patterns for
several potential outputs corresponding to the underlying form /bEd/:

---

1For a finite constraint set, there is only a finite number of possible rankings, so strictly speaking, brute-force testing for equivalence is possible: one may simply build every possible ranking and test whether the two tableaux/ERC sets are compatible with it. However, the number of logically possible rankings of n constraints is n!, so the complexity of brute-force testing is factorial in the number of constraints. This should be compared with the merely polynomial time complexity of our new test for equivalence through normalization in 16.

2My description of the Dutch pattern is based on [Kager, 1999].
According to the OT conventions, solid vertical lines in the tableau indicate that the left-to-right order of the constraints corresponds to their ranking in the grammar: \*Voiced-Obs-Coda >> Ident-Voice >> \*Voiced-Obs. The constraint \*Voiced-Obs penalizes any voiced obstruent. Its specialized cousin \*Voiced-Obs-Coda only penalizes voiced obstruents in the coda position of a syllable. Finally, Ident-Voice penalizes mismatches in voice between underlying and output consonants. The ranking in the tableau ensures that [bEt] is the winning output form: [bEd] and [pEd] lose to [bEt] in the highest constraint \*Voiced-Obs-Coda, and [pEt] loses to it in the next constraint Ident-Voice. Overall, the ranking says: “avoid voiced obstruents in the coda, but preserve them elsewhere”.

We worked through this example already knowing the ranking. Normally the work of an OT analyst proceeds in the opposite direction: she would know the constraints, the violation profiles, and the designated winner, and needs to uncover the ranking that selects the winner correctly. For that procedure, it is more convenient to use a comparative OT tableau, [Prince, 2000]. The comparative counterpart of 1 is given in 2. Each row of a comparative tableau corresponds to a pair of the winner output and one of the loser outputs of the regular OT tableau as in 1. For a specific row corresponding to a specific winner-loser pair, if the winner incurs less violations than the loser in a given constraint, the relevant cell is marked with a W; if the loser incurs less violations, the cell is marked with an L. If there is a tie, it is marked with an e.

\[
\begin{array}{|c|c|c|c|}
\hline
\text{UR: /bed/} & \text{\*Voiced-Obs-Coda} & \text{Ident-Voice} & \text{\*Voiced-Obs} \\
\hline
a. [bEd] & * & * & ** \\
b. [bEt] & * & * & * \\
c. [pEd] & * & * & * \\
d. [pEt] & & & ** \\
\hline
\end{array}
\]

It is easy to see that converting a traditional OT tableau into a comparative tableau loses information about the number of violations. But the lost information is irrelevant for recovering the ranking. Moreover, the condition on rankings that select the correct winner becomes very simple with comparative tableaux: a ranking...
selects the right winner iff in every row, all L-constraints are dominated by a W-constraint. This condition on compatible rankings is called Elementary Ranking Condition, or ERC, by [Prince, 2002]. In 2, we can see that the pair \([\text{bet}] \sim [\text{ped}]\) from 2 necessitates the inclusion of a pairwise ranking \(\text{IDENT-VOICE} \gg \ast \text{VOICED-OBS}\) into our grammar, and that the pair \([\text{bet}] \sim [\text{ped}]\) does not add any useful information: without any Ls in the row, \([\text{ped}]\) is going to lose to \([\text{bet}]\) on any possible ranking of our three constraints (i.e. the ERC corresponding to this comparative row is trivial, as it is compatible with any ranking.) In what follows, I will be largely talking in terms of comparative rows and tableaux, but it is easy to translate this into talk about ERCs and ERC sets.

The following terminology, mostly borrowed from [Prince, 2002], will be useful. A ranking \(M\) is (OT-)compatible with a comparative tableau \(T\) iff for every row, every L-constraint is dominated by some W-constraint. We also say that such \(M\) explains \(T\). If every ranking compatible with tableau \(T\) is also compatible with tableau \(U\), we say that \(T\) entails \(U\). When \(T\) and \(U\) are compatible with exactly the same rankings, they are called OT-equivalent. It is trivial to extend the notions to ERC sets.

Once a comparative tableau is computed, the actual input-output pairs are no longer needed for the task of determining the correct ranking. Thus we may freely combine several tableaux stemming from different input forms into a single bigger tableau: when the information in it is viewed as the information about the grammar of the language rather than about particular linguistic forms, we may be no longer interested in where each particular datum came from. It is with such tableaux, viewed simply as formal objects that are not given a particular linguistic interpretation, that we will be working in this paper.

Tableaux directly computed from particular linguistic forms are often suboptimal in how they represent information. E.g., the second row of 2 may be omitted without any loss of information; similarly, the W in \(*\text{VOICED-OBS}\) in the first row is “false”, because replacing it with an \(e\) will not change which ranking selects the right winner. It thus becomes important to study equivalence relations between comparative tableaux/ERC sets. To name just a few examples, [Hayes, 1997] (cf. also a follow-up in [Prince, 2006]) seeks to find transformations for tableaux allowing for better information extraction; [Prince, 2000] introduces the notion of entailment between rows and tableaux; [Brasoveanu and Prince, 2005] (published as [Brasoveanu and Prince, 2011]) define an algorithm transforming an arbitrary tableau into a small-size “basis” conveniently representing the information in the input tableau.

---

4 [Prince, 2002] introduces the logical perspective on OT compatibility wherein rows/ERCs are formulas, and rankings are essentially models. Hence \(M, N\) as designations for rankings.
The current paper continues that line of investigation. Namely, I prove that the equivalence-preserving operations introduced in the earlier literature are already enough, once we add the necessary proofs, to handle equivalence classes of comparative tableaux/ERC sets. By definition, any (comparative) tableau $T$ belongs to an equivalence class $C$ such that any tableau in $C$ is compatible with exactly the same rankings. Whenever there are such non-trivial equivalence classes, there is a problem of handling them: in geometry, there are congruence classes of geometrical figures; in proof theory, there often exist many proofs of the same statement; in lambda-calculus, there are plenty of equivalent lambda-terms. In all those cases we want to be able to obtain results common for the equivalence class. Our strategy for getting a handle on equivalent classes of OT tableaux will be fairly standard: we will find a special representative which exists in every equivalence class, and is unique in it — in other words, a normal form that can represent the class.

The plan is as follows. In Section 2 I review several elementary equivalence-preserving transformations of tableaux from the earlier literature, adding their reverses where needed. Later it will be shown that the introduced set of operations is functionally complete (that is, any equivalence-preserving transformation can be decomposed into a sequence of elementary transformations from the set). In Section 3 I define a normal form for OT tableaux, and prove the central result of the paper: normal form is unique in its equivalence class. This means that the normal form may be used as the representative of a class, or its name. Finally, in Section 4 I provide several easy corollaries following from the normal form theorems. For example, we obtain a test of equivalence for OT tableaux, and a proof that bases of [Brasoveanu and Prince, 2005]/[Brasoveanu and Prince, 2011] are unique in their equivalence classes and thus can serve as class representatives (just as normal form tableaux can).

2. Five elementary equivalence-preserving transformations

In this section, we provide the definitions for five operations with inverses that will be shown in the next section to form a functionally complete set. The operations are either trivial (3 and 4) or have been described and proven correct before (5 and 6 are either explicitly discussed by, or immediately follow from [Prince, 2002]; 7 is studied in [Prince, 2006]. The proofs of equivalence-preservation are provided here mainly for completeness’ sake, so the readers familiar with the operations may wish to skip them. The novelty of the present paper is not in the operations themselves, but in the fact that together they form a functionally complete set that is enough to represent any possible equivalence-preserving operation whatsoever.

The example tableaux are given with dashed vertical lines to emphasize that we do not presume any particular ranking — on the contrary, we seek to preserve the
ranking-equivalence in our transformations. Constraint names are chosen to be $C_1$, $C_2$, ..., rather than the usual meaningful names, to underscore the fact that the transformations are completely blind to actual linguistic data, and only concern the formal information encoded in a tableau.

We use variables $M$, $N$, ... for OT rankings; variables $T$, $U$, ... for comparative OT tableaux; and $r$ and $q$ for comparative OT rows. $W(r)$, for row $r$, denotes the set of constraints that have a $W$ in $r$. Similarly for $L(r)$. This short notation allows to define new rows compactly: e.g., if we say that $W(r) = \{C_3\}$ and $L(r) = \{C_1\}$, and $\text{Con}$ is the 5-constraint set $\{C_1, C_2, C_3, C_4, C_5\}$, then row $r$ is the row $(L, e, W, e, e)$.

The first two operations we will consider are trivial. First, row swaps, (3) never affect OT-equivalence as the order of the rows is not significant for determining whether a ranking $M$ is compatible with the tableau. (If we think in terms of corresponding ERC sets, the very concept of row order becomes irrelevant.) Second, if a tableau is not compatible with any ranking whatsoever (that is, if it puts contradictory requirements on the ranking of constraints), there is almost no useful information in it anyway, so as long as the tableau remains contradictory, any changes to it do not offend equivalence, (4).

(3) **Row swaps**: swapping any two rows preserves OT equivalence.

\[
\begin{array}{c|c|c|c|c}
C_1 & C_2 & C_3 & C_4 \\
\hline e & W & L & L \\
W & e & L & e \\
\end{array} \iff \begin{array}{c|c|c|c|c}
C_1 & C_2 & C_3 & C_4 \\
\hline W & e & L & e \\
e & W & L & L \\
\end{array}
\]

*Proof*: trivial.

(4) **Contradictory jumps**: for a contradictory tableau (that is, a tableau not compatible with any ranking), any row can be added, or, inversely, subtracted as long as the resulting tableau is still contradictory.

\[
\begin{array}{c|c|c|c|c}
C_1 & C_2 & C_3 & C_4 \\
\hline W & L & e & e \\
L & W & e & e \\
\end{array} \iff \begin{array}{c|c|c|c|c}
C_1 & C_2 & C_3 & C_4 \\
\hline W & L & e & e \\
L & W & e & e \\
e & e & W & L \\
\end{array}
\]

*Proof*: trivial.

Row splitting and its inverse, row merging, are more interesting operations. Given the ERC theory of [Prince, 2002], it is trivial to show that a row with several $L$s is equivalent to a set of single-$L$ rows. In ERC terms, such single-$L$ rows have been called Primitive Ranking Conditions by [Prince, 2006, p.4]. The correctness of row splitting and row merging shows that explaining each $L$ in a multiple-$L$ row is in a sense independent from explaining the other $L$s. Working with single-$L$ rows, or
PRCs, is often more convenient, especially when we turn all rows in a tableau into this single-L/PRC form.

(5) **Row splittings and mergings:** a row \( r \) is equivalent to any set of rows \( r_1, \ldots, r_n \) such that \( \forall r_i : W(r_i) = W(r) \), and \( \bigcup_i L(r_i) = L(r) \). That is, \( r, r_1, \ldots, r_n \) must have exactly the same Ws, and the combined Ls of \( r_1, \ldots, r_n \) must form the same set as the Ls of \( r \).

\[
\begin{array}{cccc}
C1 & C2 & C3 & C4 \\
W & W & L & L \\
\end{array}
\quad \Rightarrow \quad \\
\begin{array}{cccc}
C1 & C2 & C3 & C4 \\
W & W & L & e \\
W & W & e & L \\
\end{array}
\]

**Proof:** Suppose a ranking \( M \) puts on top of each L in \( r \) one of \( r \)'s Ws. As any \( r_i \) has the same Ws, any L in any \( r_i \) will also be covered by a W under ranking \( M \).

Conversely, suppose a ranking \( N \) is compatible with all rows \( r_1, \ldots, r_n \). Consider some L of row \( r \). Some \( r_i \) must have an L in the same constraint, and \( N \) covers it by a W in one of \( W(r_i) \). That W-constraint in \( r_i \) also has a W in \( r \), by definition. Thus \( N \) covers the arbitrary L in \( r \) just as well.

Thus a ranking is compatible with \( r \) iff it is compatible with \( r_1, \ldots, r_n \). \( \square \)

The remaining two pairs of operations are most complex in the set. Some OT rows may be superfluous in their tableaux: even if we delete them, the amount of information in the tableau does not change. (E.g., the second row in 2 is superfluous.) By definition, subtraction or addition of such rows does not offend OT equivalence. What is non-trivial, though, is determining the exact formal conditions under which a row is superfluous. In the proof, I use the criterion by \cite{Prince2002}, featuring his operation of fusion \footnote{The operation of fusion on rows is defined in \cite{Prince2002}'s (12), p. 8. For tableau \( U \), the fusion row \( fU \) has an \( e \) in the \( Ci \) cell iff all rows in \( U \) have an \( e \) in \( Ci \); has an L iff some row in \( U \) has an L in \( Ci \); and has a W otherwise, that is, when at least one row in \( U \) has a W in \( Ci \), and all other rows have either Ws or es, but not Ls.}

There exists an alternative characterization of superfluousness based on domination chains of constraints, but the proof based on such chains is more cumbersome \footnote{I provide the definitions of possible and maximal domination chains in (i), and the criterion of superfluousness based on them, without proof, in (ii):

(i) For a tableau \( T \), a row \( r_i \in T \), and a \( C_j \in L(r_i) \), a **possible domination chain** is a sequence of constraints \( \langle C_{k_1}, \ldots, C_{k_n} \rangle \) s.t. \( C_{k_n} = C_j \), a single constraint never occurs twice in the chain, and for each \( C_{k_i}, C_{k_{i+1}} \), there is a row \( r_m \in T \) for which \( C_{k_i} \in W(r_m) \), \( C_{k_{i+1}} \in L(r_m) \). A **maximal possible domination chain** is a possible domination chain for which there is no \( r_m \in T \) s.t. \( C_{k_i} \in L(r_m) \).

(ii) **Superfluous row theorem.** A tableau \( T = \langle r_1, \ldots, r_n \rangle \) entails a row \( q \) iff for each \( Ci \in L(q) \), there exists such a row \( r \in T \) in every maximal domination chain for \( Ci, r, \) and \( T \), that there is a constraint \( C_{k_i} \) in it s.t. \( C_{k_i} \in W(q) \).}
Using the fact that row order is not significant in a tableau, cf. 3, we can safely use set notation for tableaux, understood as being parasitic on the notation for ERC sets: \( T \setminus r \) denotes tableau \( T \) with row \( r \) subtracted; \( T \cup U \) is a concatenation of tableaux \( T \) and \( U \); and so forth.

(6) **Inference eliminations and introductions**: a row \( r \) entailed in tableau \( T \) by the rest of the tableau (that is, by \( T \setminus r \)) can be subtracted from \( T \), or added back to tableau \( T \setminus r \).

\[
\begin{array}{cccc}
C1 & C2 & C3 & C4 \\
W & L & e & e \\
e & W & L & e \\
W & e & L & e
\end{array} \quad \Leftrightarrow \quad \begin{array}{cccc}
C1 & C2 & C3 & C4 \\
W & L & e & e \\
e & W & L & e \\
W & e & L & e
\end{array}
\]

**Proof**: Trivial. What is non-trivial is how to determine if \( r \) is entailed by \( T \setminus r \). By Prop. 2.5 of [Prince, 2002], \( r \) is entailed by \( T \setminus r \) iff there exists a subtableau \( U \) of \( T \setminus r \) s.t. the fusion (cf. fn. 5) of \( U \) is a row \( q \) with \( W(q) \subseteq W(r) \) and \( L(q) \supseteq L(r) \). \( \square \)

To reduce computational complexity, an RCD-based method is proposed by [Prince, 2002, Sec. 5]. Prince shows that instead of fusion of all subtableaux, one may check whether \( T \setminus r \) is consistent with the negative \( \neg r \) of \( r \), obtained by replacing all \( r \)'s Ws with Ls and vice versa. For \( m \) rows, we need \( m \) such RCD-based checks. As [Magri, 2009] explains, RCD requires \( m^2n \) operations for a tableau with \( m \) rows and \( n \) constraints. The complexity of RCD-based inference elimination is thus polynomial, in contrast to subtableau-fusion version which is exponential in the number of constraints \( n \).

Finally, not all Ws in an OT tableau are necessarily equal: there may be rows with “false Ws” such that there is no ranking compatible with the tableau which puts that W on top of any Ls in the row. As shown by [Prince, 2006, p. 12], such false Ws may be replaced with an e without affecting the set of rankings the tableau is compatible with. The operation for doing that is called **Generalized Removal of W**, or GRW. We also introduce its inverse, **Generalized Introduction of W**, or GIW.

(7) **Generalized Removal of W (GRW) and Introduction of W (GIW)**: informally, a “false” W is a W whose replacement with an e does not change which rankings the tableau is compatible with. Thus a false W does not do any actual work. The example tableaux below may help to visualize the phenomenon.

---

Checking the criterion based on maximal chains does not require computing new rows, as the fusion criterion does. But it is easy to see from the cumbersomeness of the definitions that proving the criterion’s correctness from first principles requires a bit of work. Therefore I simply reuse Prince’s fusion-based result in the main text, referring the reader to [Prince, 2002] for proofs of its correctness.
Turning to the formal definition, for rows \( r \) and \( r' \) such that instead of \( r \)'s W in a fixed \( C_i \), row \( r' \) has an e, consider a pair of \( T \) including \( r \) that is not entailed by the rest of \( T \), and \( T' := (T \setminus r) \cup r' \) (that is, \( T' \) that is exactly like \( T \), but with the W in \( C_i \) in row \( r \) replaced with an e).

The claim is: \( T \) and \( T' \) thus defined are equivalent iff \( T \setminus r \) entails the row \( q \) such that \( W(q) := L(r) \cup (W(r) \setminus C_i) \), and \( L(q) = \{C_i\} \).

**Proof:** Just as with inference eliminations, the fact that a false W can be replaced with an e is trivial. What is non-trivial is the criterion for false Ws: a W is false iff the row \( q \) as described above is entailed by \( T \setminus r \). [Prince, 2006] proves essentially that criterion in his (31) using fusion. [Author, 2011] provides a different proof in their (125) using partial OT rankings. The proof below is based on the idea of the proof in [Author, 2011], but does not use either fusion or the apparatus of partial rankings.

Consider row \( r \) with a W in \( C_i \), and row \( q \) defined as in the criterion above: \( W(q) \) contains all W- and L-constraints of \( r \) except \( C_i \), and the only L-constraint of \( q \) is \( C_i \). We need to show that the W in \( C_i \) in \( r \) is false in tableau \( T \) precisely when the rest of the tableau, \( T \setminus r \), entails the row \( q \) so constructed.

Without loss of generality, assume that \( r \) has only one L, in constraint \( C_j \). (We have the right to assume that because we proved in 5 that any multiple-L row may be split into several single-L rows that are together equivalent to it.)

Suppose \( q \) is entailed by \( T \setminus r \). We will prove that \( T \) with \( q \) is then equivalent to \( T' \), and thus the W \( C_i \) in \( r \) is false. Assume towards a contradiction that there is a ranking \( M \) which is compatible with \( T \), but not with \( T' \). That ranking \( M \) must be compatible with \( r \), but not with \( r' \) which differs from it in that it has an e in \( C_i \) instead of a W. Then \( M \) must say that \( C_i \gg C_j \) and that for every \( C_k \) from \( W(r) \setminus C_i \), \( C_j \gg C_k \): otherwise it would be compatible not only with \( r \), but also with \( r' \). But then \( M \) is incompatible with \( q \): the L-constraint \( C_i \) dominates \( C_j \) in \( M \), and then by transitivity any W-constraint \( C_k \). This is contrary to assumption, and therefore there cannot be such an \( M \). Furthermore, any ranking compatible with \( T' \) is bound to be compatible with \( T \), and thus we derive that if \( q \) is entailed by \( T \setminus r \), then \( T \) and \( T' \) are OT-equivalent.

For the other direction, suppose \( q \) is not entailed by \( T \setminus r \). We show that then there is a ranking compatible with \( T \), but not \( T' \). We need to show that there exists ranking \( M \) compatible with \( T \setminus r \) that says \( C_i \gg C_j \gg C_k \) for all \( C_k \in W(r') \): such a ranking will be compatible with \( r \), but not with \( r' \). Towards a contradiction, suppose there is no

---

7Prince’s theorem is slightly weaker compared to our formulation: Prince requires all rows in \( T \) to be not entailed by the rest of the tableau. His actual proof, though, only employs the fact that \( r \) is not entailed by \( T \setminus r \), just as our proof does.
such $M$. That is only possible if no ranking compatible with $T \setminus r$ says $Ci \succ Cj \succ Ck$. For the $Ci \succ Cj$ part, $T \setminus r$ cannot necessitate the opposite ordering $Cj \succ Ci$: if it did, then it would have entailed $q$, contrary to assumption. For $Cj \succ Ck$, suppose towards a contradiction that every ranking compatible with $T \setminus r$ says for some $Ck \in W(r')$ or other that $Ck \succ Cj$. That can only be if there is a row $s$ in $T \setminus r$ with an L in $Cj$, and $W(s) \subseteq W(r')$. But if that is so, then the row $s$, and thus $T \setminus r$ as a whole, entail $r$: the L is in the same place in $s$ and $r$, and $W(s) \subseteq W(r') \subset W(r)$. That is contrary to assumption, so if $T \setminus r$ does not entail either $q$ or $r$, then there must be a ranking $M$ compatible with $T \setminus r$ saying $Ci \succ Cj \succ Ck$ for all $Ck \in W(r')$. That $M$ is compatible with $T$, but not with $T'$, and thus witnesses that $T$ and $T'$ are not OT-equivalent: $T$ is compatible with a larger number of rankings, thanks to the non-false W for which the criterion based on a specially constructed row $q$ fails. □

We have now defined and proved correctness of five pairs of elementary operations on comparative OT tableaux preserving OT-equivalence. Those operations as such have been known before. What has not been known is that those five pairs form a functionally complete set: any transformation preserving OT-equivalence can be performed by applying a sequence of those elementary operations, as we will show in the next section.

The following easy-to-prove fact will become useful later:

(8) All operations in 3-7 have inverses: row swap is self-inverse; for the other four pairs, the two members of the pair are inverses.

What (8) means is that each sequence of applications of our elementary operations may be inverted: if we can derive from tableau $T$ another tableau $U$ using those operations, then we can also derive from $U$ the original tableau $T$ by applying the inverted form of the same sequence.

3. Normal form for OT tableaux

In this section, we present the core novel results of this paper: two theorems regarding the existence and uniqueness of normal form for comparative OT tableaux. Namely, we define a special tableau format in (9) and then prove that for each equivalence class $C$, there exists exactly one tableau in such format, and moreover, that the normal form of a (finite) tableau is computable. Normal forms thus can serve as true representatives of their equivalence classes, giving us a handle on those.

It should be stressed that there is nothing particularly special about normal forms — in fact, as we will see in the next section, other forms may be proven to be usable as normal forms just as well. The reason we define the normal form in (9) in this way is simply that it is convenient for proof purposes.

(9) Normal form for OT tableaux:
(1) The only contradictory tableau in the normal form is the one-row tableau with a single L in the first constraint.

(2) Each row has at most a single L (=corresponds to a PRC ERC).

(3) There are no rows which can be inference-eliminated (see [6]).

(4) In multiple-W rows, there are no false Ws (see [7]).

(5) The rows are ordered according to some strict total order of the set of all possible rows. (For corresponding ERC sets, the notion of row order becomes irrelevant.)

Here is an example of a normal form tableau:

```
<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
<th>C5</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>e</td>
<td>L</td>
<td>e</td>
<td>e</td>
</tr>
<tr>
<td>W</td>
<td>e</td>
<td>e</td>
<td>L</td>
<td>e</td>
</tr>
<tr>
<td>W</td>
<td>e</td>
<td>e</td>
<td>e</td>
<td>L</td>
</tr>
<tr>
<td>e</td>
<td>W</td>
<td>e</td>
<td>e</td>
<td>L</td>
</tr>
<tr>
<td>e</td>
<td>W</td>
<td>e</td>
<td>L</td>
<td>e</td>
</tr>
</tbody>
</table>
```

Just calling something a normal form does not make it one. The results in [11][14] establish the fact that the class of tableaux defined in [9] indeed has normal form properties.

(11) **Normal Form Existence Theorem.** An arbitrary (finite) tableau $T$ can be transformed into an equivalent normal form tableau by a (finite) sequence of equivalence-preserving transformations in [3][7].

(12) **Corollary to [11]** Each non-empty equivalence class of tableaux contains at least one normal form tableau.

**Proof of [11]**. We give an explicit procedure for transforming an arbitrary tableau so that it satisfies the requirements in [9]. For contradictory tableaux, we just add the row (L, e, e, ...), and subtract all others. If the tableau is not contradictory, we apply row splittings until all rows have at most one L (=are turned into PRCs). Assuming the tableau is finite, we can eliminate all entailed rows by testing if fusions of subtableaux satisfy Prince’s condition on entailment, see [6]. After that, we can similarly eliminate all false Ws from the resulting tableau by testing if the conditions for GRW [7] are met (as all entailed rows were eliminated by that point, the row independence precondition of the criterion in [7] is met). We finish the procedure by applying row swaps to get the ordering right.

**Proof of [12]** Trivial: if there were no normal form tableau in a non-empty equivalence class, then the theorem [11] could not have been valid.

The actual choice of ordering is irrelevant as long as all conceivable rows are strictly ordered. I will use the following: 1) let the first constraint where only one of $r$ and $q$ has a W be $Ct$; then the row with the W in $Ci$ goes first; 2) for rows which have identical W-sets, the row which has an L in the first constraint where only one of them has an L goes first.
Note that if the tableau is finite, the normalization procedure described in our proof of (11) is computable. This is important because if the normal form were not computable, we would not be able to use it without restrictions in place of any other tableau in its equivalence class as we cannot ensure we can actually derive one from the other in a finite amount of time.

It may seem natural that the OT normal form is computable, yet in fact it is a non-trivial result: many innocent-looking problems are not computable, while their seeming close cousins are. For instance, it can be determined in finite time (= it is computable) whether an arbitrary sentence of modal logic is provable, but the same problem for first-order logic is not computable. So it is a significant result that each OT tableau may be computably converted into the normal form. Moreover, complexity analysis shows that normalization is quite efficient:

(13) Normalization as defined in the proof of (11) runs in time polynomial in the number of rows \( m \) and the number of constraints \( n \).

Proof of (13) Consider tableau \( T \) with \( m \) rows and \( n \) constraints. Consistency check may be performed through fusing all subtableaux of \( T \) and checking if any resulting fused row has only Ls — or equivalently and faster using RCD, as shown by [Prince, 2002, Sec. 4]. To perform RCD, we need \( m^3 n \) operations, [Magri, 2009, p. 371]. Next, we do row splittings, which for any of the \( m \) rows cannot result in creating more than \( n \) rows of \( n \) constraints each, so this requires at most \( m n^2 \) operations. The number of rows in the resulting split tableau is not greater than \( mn \). Next, we check for entailed rows to eliminate. As we discussed above regarding (6) rather than doing subtableau fusion, exponential in the number of rows \( m \), we can do instead \( m \) RCD-based checks as described by [Prince, 2002, Sec. 5]. We have \( mn \) rows, and each RCD involves \( (mn)^3 n \) operations, so overall we need \( m^3 n^4 \) operations for this step. Finally, we need to check for false Ws. For that we check every W in \( mn \) rows, so at most this would be \( mn^2 \) checks (actually, much less, as the same row cannot contain both \( n \) Ls and \( n \) Ws, but we can ignore this.) Each test involves checking whether the rest of the tableau entails a specially constructed row for each particular W. Again, an entailment check costs for a single row and a tableau with \( mn \) rows is \( m^3 n^4 \), so overall we have at most \( m^4 n^6 \) operations. This will be the dominating term in our complexity estimate. The time complexity of normalization is thus polynomial, i.e. very good. \( \square \)

From (11) and (12) we know that each equivalence class has at least one normal form tableau. But can a class contain more than one normal form in it? Theorem (14) shows that it cannot, and thus a normal form tableau defines its class: it is its unique representative. To prove that fact, we will need to use relatively complex ranking-construction techniques.

(14) Normal Form Uniqueness Theorem. In each equivalence class of OT tableaux, there is at most one normal form tableau.
Proof of [14] We show that any two distinct normal form tableaux $T$ and $U$ belong to different equivalence classes.

Pick some row $r$ from $T$ which is not shared by $U$ (in case $T \subset U$, we immediately derive the conclusion by considering a row from $U$ that is not in $T$, and the fact that $T$ cannot entail that row). Either our pick $r$ is entailed by $U$, or it is not. In case $r$ is not entailed by $U$, there is some ranking $M$ compatible with $U$, but not with $r$, and thus not with $T$, so $U$ and $T$ are not OT-equivalent.

The interesting case is when $U$ entails the row $r$ we picked. We will show that in that case, there must be some ranking compatible with $T$, but not with $U$. We pick a minimal subtableau $V$ of $U$ that still entails $r$. As $V$ entails $r$, every ranking compatible with $V$ must also put one of $r$’s Ws on top of $r$’s L. That can only be if there is a row $q \in V$ which has an L in the same constraint where $r$ has an L. Let’s call that constraint $Ci$.

Suppose towards contradiction that $T$ and $U$ are equivalent, that is, compatible with exactly the same rankings. Consider some $M$ compatible with $T$, and accounting for $V$ “in the minimal possible manner”: let $M$ contain the domination chain $Ck_1 \gg Ck_2 \gg \ldots \gg Ci$ where each pairwise ranking $Ck_1 \gg Ck_2$, ..., $Ck_n \gg Ci$ accounts for one of the rows in $V$, but no other pairwise rankings accounting for any of $V$’s rows. As $V$ is in normal form and all its Ws are not false, it must be possible to construct such an $M$. As $V$ entails $r$, constraint $Ck_1$ is a W-constraint in $r$.

$T \setminus r$ cannot entail $V$: if it did, it would have entailed $r$ by transitivity, which is contrary to the normal form assumption. Therefore it must be possible to lower one of $Ck_i$ constraints below $Ci$ building a ranking $M'$ which is still compatible with $T \setminus r$, but not with $V$. As $V$ entails $r$ by assumption, $M'$ must also be incompatible with $r$. But that can only be if the lowered constraint has to be $Ck_1$, a W-constraint in $r$, for otherwise $M'$ would have still said $Ck_1 \gg Ci$.

We modify $M'$ as follows: raise $Ck_1$ just on top of $Ci$, but below $Ck_n$, resulting in $M'' = \ldots Ck_2 \gg Ck_3 \gg \ldots \gg Ck_n \gg Ck_1 \gg Ci$. That ranking $M''$ is incompatible with $V$, because by construction there must have been a row in $V$ for which we needed the pairwise ranking $Ck_1 \gg Ck_2$, and $M''$ says $Ck_2 \gg Ck_1$. But at the same time $M''$ is compatible with $r$, as it puts one of its Ws on top of its L. Now compare $M'$ and $M''$, and consider their compatibility with $T \setminus r$. $M'$ was compatible with $T \setminus r$. $M''$ differs from it in that it says $Ck_1 \gg Ci$ instead of $Ci \gg Ck_1$. That change could not make $M''$ incompatible with $T \setminus r$: the initial ranking $M$ also said $Ck_1 \gg Ci$ and was compatible with $T \setminus r$. Therefore we have built a ranking, namely $M''$, which is compatible with $T \setminus r$ and with $r$, but not with $V$. This ranking witnesses that $T$ and $U$ are not equivalent. 

[11] and [14] together entail that there is exactly one normal form tableau per equivalence class. Thus a tableau as described in [2] is a true normal form: a full-fledged representative, or a “name”, of its equivalence class.

In practical terms, that means that in our proofs, we can capitalize on the many nice properties of normal forms, knowing that the results will generalize to arbitrary
tableaux. In the next section, we illustrate the use of the normal form results in several simple corollaries.

4. Capitalizing on the normal form results

The statements in 15, 16 and 18 serve two purposes. First, they have independent value, especially the proof that Brasoveanu and Prince’s SKB bases are unique in their equivalence classes. ([Brasoveanu and Prince, 2011] claim to have proven, in an unpublished manuscript, a much weaker fact that for a given tableau, the SKB basis is unique. 18 shows that in fact for any equivalent tableaux the SKB has to be the same.) Second, the proofs of these statements illustrate how one can use the normal form results in practice to handle equivalence classes of OT tableaux.

(15) The operations in 3-7 form a functionally complete set: any tableau can be transformed into any equivalent tableau by a sequence of such operations.

Proof of 15. By 11 and 14, any pair of equivalent tableaux may be transformed into the same normal form tableau by a sequence of operations in 3-7. To conclude the proof, we observe that an inverted sequence transforms the normal form back into the original tableau. By normalizing the first tableau, and then de-normalizing it by applying the inverted sequence built for the second tableau, we transform the first tableau into the second. □

(16) Equivalence of finite OT tableaux is computable in polynomial time.

Proof of 16. To test tableaux T and U for equivalence, it suffices to normalize both and check whether the resulting normal forms are the same. All operations are computable, for finite tableaux.

The complexity of this test is polynomial: by 13, the time complexity of normalization is polynomial in the number of rows m and the number of constraints n, and we need two such normalizations, plus a comparison of two resulting normal form tableaux which is also polynomial in m and n for the original tableaux. This fairly moderate complexity may be compared with the enormous factorial complexity of the brute-force test for equivalence that involves testing every possible ranking for compatibility with each tableau. □

[Brasoveanu and Prince, 2005][Brasoveanu and Prince, 2011] define a dense format of tableaux called the Skeletal Basis (SKB), and an algorithm turning an arbitrary tableau into an equivalent tableau in that format. An SKB of tableau T is a tableau T’ such that 1) there is no OT-equivalent tableau with a fewer number of rows; and 2) no other equivalent tableau of the same cardinality has more es. 17 is the Skeletal Basis of the normal form tableau in 9.
Brasoveanu and Prince claim (citing an unpublished proof in manuscript [Prince and Brasoveanu, 2010]) that each tableau has a unique SKB. We prove a much stronger result in [18]: each equivalence class of tableaux has a unique SKB.

(18) Each equivalence class of OT tableaux has exactly one tableau in the Skeletal Basis (SKB) form of [Brasoveanu and Prince, 2011].

Proof of [18] By showing that SKB-s are in one-one correspondence with normal forms.

If we apply all possible row mergers to a normal form tableau, we get an SKB: the original normal form tableau did not have superfluous rows, so the quantity of the rows in the resulting tableau will be minimal; furthermore, as the normal form tableau does not contain any false Ws, the resulting tableau will have the maximal number of es.

In the other direction, if we split all rows of an SKB into one-L rows, there can be no superfluous rows in the result (otherwise the L corresponding to a superfluous row could have been replaced with an e, contrary to the definition of an SKB which must have as many es as possible); as for false Ws, there can be none in the SKB tableau itself, and after all rows splittings are applied, no new false Ws can arise (if a false W could arise in one of the resulting one-L rows, then the same W would have been false even before splitting).

What remains is to show that there can be no two SKB-s in the same equivalence class. Suppose towards contradiction there are two SKB-s $S_1$ and $S_2$. They both normalize to the same normal form tableau by the procedure above. From the definition of SKB, only row splittings are required. Pick an arbitrary set of rows $r_1, ..., r_n$ with Ws in the same constraints from the resulting normal form tableau. If $S_1$ and $S_2$ each have only one row splitting into this same set, that must be the same row. If $S_1$ and $S_2$ have more than one row splitting into this set, we can actually merge those rows into just one, resulting in a smaller equivalent tableau $S_3$, contrary to the assumption of $S_1$ and $S_2$’s minimality. Thus either $S_1 = S_2$, or they are not minimal possible size in their equivalence class. Therefore there is only one SKB per class. □

[18] essentially means that all nice results about normal forms may be transferred to SKB-s. For instance, the equivalence test in [16] may be replaced by an equivalence test comparing SKB-s derived using Brasoveanu and Prince’s Fusional Reduction algorithm. With [18] in hand, we may employ Brasoveanu and Prince’s SKB-s as representatives of their equivalence classes instead of our normal forms. Normal forms are often more convenient in complex proofs, because the relations between constraints in them are maximally untangled; but SKB-s are more useful when it becomes convenient to have smaller-sized representatives.
5. Conclusion

We defined a normal form for OT tableaux, and showed that there is exactly one normal form in each equivalence class of OT tableaux. Moreover, we have demonstrated that each OT tableau can be computably normalized by a sequence of five pairs of previously known equivalence-preserving transformations in 3\(^7\). The computational cost of normalization is only polynomial in the number of rows \(m\) and constraints \(n\), thanks to the use of the efficient RCD-based algorithm for entailment checking proposed by [Prince, 2002] Sec. 5.

Those results provide us with a handle on equivalence classes of OT tableaux: using them, we may reason about tableaux without any loss of generality while only considering normal forms. The several examples in Section 4, including the fact in 18 that Brasoveanu and Prince’s Skeletal Bases are unique in their equivalence classes, illustrate how to capitalize on the presented OT normal form theorems.

References