Hierarchical Permutation Complexity for Word Order Evaluation

Miloš Stanojević **Khalil Sima'an**
Institute for Logic, Language and Computation (ILLC)
University of Amsterdam
{initial.last}@uva.nl

Abstract

Existing approaches for evaluating word order in machine translation work with metrics computed *directly* over a *permutation* of word positions in system output relative to a reference translation. However, every permutation factorizes into a permutation tree (PET) built of *primal permutations*, i.e., atomic units that do not factorize any further. In this paper we explore the idea that permutations factorizing into (on average) *shorter primal permutations* should represent *simpler ordering* as well. Consequently, we contribute *Permutation Complexity*, a class of metrics over PETs and their extension to forests, and define *tight metrics*, a sub-class of metrics implementing this idea. Subsequently we define example tight metrics and empirically test them in word order evaluation. Experiments on the WMT13 data sets for ten language pairs show that a tight metric is more often than not better than the baselines.

1 Introduction

MT evaluation involves at least two factors, word order (syntactic) and adequacy (semantic). Conceivably, MT system developers could use diagnostic tools based on metrics dedicated to each factor separately. Word order metrics are frequently used to evaluate pre-ordering components, e.g., (Herrmann et al., 2011; Bisazza and Federico, 2013), or for analyzing specific reordering phenomena, e.g., (Bisazza and Federico, 2013; Xiang et al., 2011; Braune et al., 2012). Other uses include, ordering component tuning, e.g., (Gao et al., 2011; Neubig et al., 2012; DeNero and Uszkoreit, 2011; Katz-Brown et al., 2011; Hall et al., 2011), measuring divergence between languages (Birch et al., 2008), and matching gene sequences in bioinformatics (Eres et al., 2004).

For evaluating word order, a permutation is induced between a system output and the corresponding reference translation. Existing work uses metrics over permutations such as Kendall's tau (Lapata, 2006; Birch and Osborne, 2011), Spearman (Isozaki et al., 2010), Hamming, Ulam (Birch et al., 2010) and Fuzzy Score (Talbot et al., 2011). Approximately, Kendall's tau, Spearman and Hamming measure correct individual position or correct relative pairs, whereas Ulam and Fuzzy Score measure monotone units (contiguous or not).

A word order metric measures how similar a permutation is to the monotone (or identity) permutation. Here we advocate the idea that a suitable metric must *also* assign similar values to similar permutations. Crucially, factorizing a permutation into a Permutation Tree (PET) reveals its atomic building blocks, called *primal* permutations (Albert and Atkinson, 2005; Gildea et al., 2006). In this view, permutations that factorize into similar PETs should be similar. Some previous work (Stanojević and Sima'an, 2014a; Stanojević and Sima'an, 2014b) has used PETs for evaluation, but without attempting to explain the effect of factorization. Next we motivate the idea that, all other things being equal, *the more factorizable a permutation the simpler it is* in terms of ordering.

Informally, a PET for permutation π is a tree where the nodes are labeled with *operators* (Figure 1). The fringe of every subtree in a PET is a *sub-permutation* of π, i.e., a contiguous sub-sequence isomorphic with a permutation.[1] Consider $\pi_a = \langle 6, 1, 4, 2, 3, 5 \rangle$ and $\pi_b = \langle 6, 1, 5, 2, 3, 4 \rangle$. Their PETs (two

[1] Akin to a phrase pair in MT.

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 2164–2173, Osaka, Japan, December 11-17 2016.

left-most in Figure 1) are built from monotone $\langle 1,2 \rangle$ or inverted $\langle 2,1 \rangle$ operators only. Two local inversions $\langle 2,1 \rangle$ could turn each of π_a and π_b into monotone. Permutation $\pi_c = \langle 2,4,5,6,1,3 \rangle$ (right-most Figure 1) demands $\langle 2,4,1,3 \rangle$ at the root to bring it to monotone. In contrast, $\pi_d = \langle 6,2,4,1,5,3 \rangle$ does not yield to factorization because it does not properly contain sub-permutations; Non-factorizable permutations are called *primal permutations*,[2] and they constitute the *atomic building blocks for all permutations* (Albert and Atkinson, 2005) – see Section 3. Hence, π_d demands itself to convert it into monotone. In this view, π_a and π_b signify potentially simpler ordering than π_c, which is simpler than π_d.

Conveniently, PETs show two aspects of permutations: *recursive grouping and primal building blocks*. In this paper we introduce *Permutation Complexity*, a class of metrics over PETs, exploiting hitherto untapped discerning properties of permutations. **A.** Similarity: different permutations often share primal permutations. **B.** Factorizability: some permutations factorize into shorter primal permutations but others do not. **C.** Hierarchy: factorizing permutations exposes their hierarchical grouping. Practically speaking, metrics over PETs should be attractive because they parameterize in terms of primal permutations and bracketing structure.

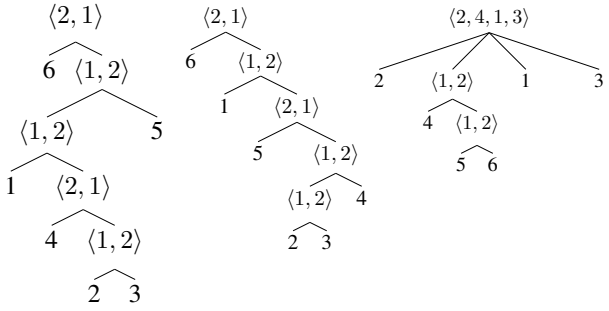


Figure 1: Three permutations and their PETs

From our Permutation Complexity viewpoint we see PET factorization as *compression* using a *code book* of primal permutations. Consequently, we introduce *tight* metrics, a sub-class that assigns a smaller complexity to a PET than to any less factorized structure of the same permutation, with the intuition that more factorization should reveal simpler building blocks. In this paper, we contribute: (1) Foundational formalization of tight complexity metrics over permutations, (2) An extension of PETs (Gildea et al., 2006) to forests to capture the potential relevance of brackettings for evaluation, (3) Novel metrics for reordering evaluation, and (4) Experiments on system ranking in MT. Our experiments show that the new tight (and semi-tight) metrics perform competitively over a range of language pairs, which provides the first evidence for a complexity-based factor in evaluation.

2 Existing metrics (Baselines)

We define evaluation metrics in the range $[0,1]$ with the interpretation *the higher the score the better*. Whilst this is natural for MT evaluation, for a formal treatment of complexity, as in Section 4, it is natural that complexity is interpreted as "the higher the more complex". The two notions are easily converted to each other after normalization.

A permutation π over $[1..n]$ (subrange of the positive integers) is a bijective function from $[1..n]$ to itself. To represent permutations we will use angle brackets as in $\langle 2,4,3,1 \rangle$. Given a permutation π over $[1..n]$, the notation π_i $(1 \leq i \leq n)$ stands for the integer in the i^{th} position in π; $\pi(i)$ stands for the index of the position in π where integer i appears; and π_i^j stands for the (contiguous) sub-sequence of integers $\pi_i, \ldots \pi_j$. The *length* of π is simply $|\pi| = n$.

$$\textsc{Kendall}(\pi) = \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\delta[\pi(i)<\pi(j)]}{(n^2-n)/2}$$

$$\textsc{Hamming}(\pi) = \frac{\sum_{i=1}^{n}\delta[\pi_i=i]}{n}$$

$$\textsc{Spearman}(\pi) = 1 - \frac{3\sum_{i=1}^{n}(\pi_i-i)^2}{n(n^2-1)}$$

$$\textsc{Ulam}(\pi) = \frac{LCS(\pi,\textsc{Id}_1^n)-1}{n-1}$$

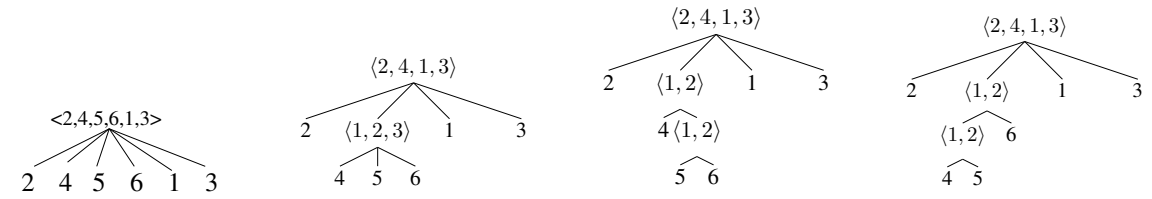
$$\textsc{Fuzzy}(\pi) = 1 - \frac{c-1}{n-1}$$

where c is # of monotone sub-permutations

Figure 2: Common metrics over permutations

The baselines are the existing metrics over permutations, including KENDALL's tau, HAMMING and ULAM used in (Birch and Osborne, 2010; Birch and Osborne, 2011; Birch et al., 2010; Isozaki et al., 2010); SPEARMAN rho used in (Isozaki et al., 2010); and FUZZY *Reordering Score* used in (Talbot et al., 2011), which is a reordering measure extracted from

[2] Also known as simple or non-decomposable (Brignall, 2010) – note the analogy with prime numbers.



(a) Original permutation (b) Factorization step (c) PET (d) Another PET

Figure 3: Permutation factorization leading to PETs

METEOR (Denkowski and Lavie, 2011). Figure 2 lists the definitions of these metrics. In these definitions, LCS stands for Longest Common Subsequence, Kronecker $\delta[a]$ which is 1 if $(a = true)$ else zero, and $\text{ID}_1^n = \langle 1, \cdots, n \rangle$ which is the identity permutation over $[1..n]$. Next we present an alternative view of permutations.

3 Factorization and order complexity

In factorization we seek to decompose a permutation to reveal a tree of its atomic ordering patterns. Figure 3 shows the factorization process applied to $\pi = \langle 2, 4, 5, 6, 1, 3 \rangle$. It starts out by representing π as a tree with root decorated with π itself (Figure 3a). In every step we seek the *minimal number* of adjacent *sub-permutations*. For $\langle 2, 4, 5, 6, 1, 3 \rangle$ this minimal number is four, namely $\{2\}$, $\{4, 5, 6\}$, $\{1\}$ and $\{3\}$. The first step leads to Figure 3b, where the sub-permutations are represented as subtrees with roots decorated with operators (permutations) over their child nodes. Applying factorization recursively to $\langle 4, 5, 6 \rangle$ leads to choices in binarization because both $\{4, 5\}$ and $\{5, 6\}$ are sub-permutations (Figures 3c and 3d). Next we summarize the formal results underlying factorization.

Primal permutations[3] are permutations that do not properly contain sub-permutations. Example common primal permutations are $\langle 1, 2 \rangle$, $\langle 2, 1 \rangle$ and $\langle 2, 4, 1, 3 \rangle$. Primal permutations signify the *atomic reorderings*. The following result shows they are also the *building blocks* of all permutations.

Factorization (Albert and Atkinson, 2005) Every permutation π can be written[4] as $\sigma[\pi_1, \cdots, \pi_m]$, where σ is *primal and unique*, and each π_j is a sub-permutation of π. If $m \geq 4$ then $\pi_1, \cdots, \pi_m$ are *unique*.

The uniqueness of σ and $\pi_1, \cdots, \pi_m$ for $m \geq 4$ is crucial for efficiency (Section 5). We call m the **arity** of π, written $\mathbf{a}(\pi)$ (or simply $\mathbf{a}$). For example, $\langle 4, 2, 3, 1 \rangle$ has arity 2: $\sigma = \langle 2, 1 \rangle$, $\pi_1 = \langle 4, 2, 3 \rangle$ and $\pi_2 = \langle 1 \rangle$. Applying Albert & Atkinson's result recursively factorizes π into PETs, see (Gildea et al., 2006) and Section 5 for efficient algorithms.

Permutation complexity is the class of metrics over PETs. This class includes *ground metrics* over primal permutations (operators), and higher-order metrics over PETs for other factorizable permutations. In a trivial sense, useful here, the existing baseline metrics can be seen as operating over weaker kinds of factorization which leave (parts of) π unfactorized.

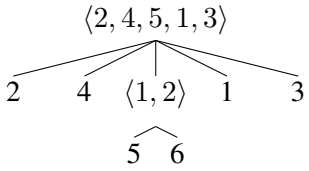


Figure 4: Weak factorization

Weak factorization A permutation π is a *weak factorization (WF)* of itself, represented as a single node with operator equivalent to π. The process applies recursively factorizing an operator in a given weak factorization τ into any number of sub-permutations (not necessarily minimal). Weak factorization may terminate at any point.

Intuitively, we would like permutation complexity metrics to be sensitive to factorization into primal permutations. This can be achieved by imposing a partial order over the different weak factorizations

[3] Also known as simple or non-decomposable permutations.

[4] The notation $\sigma[\pi_1 \cdots \pi_m]$ stands for a sequence of sub-permutations $\pi_1 \cdots \pi_m$ which is permuted by σ.

of the same permutation, assigning minimal complexity to PET factorizations. Next we formalize the notion of *tight metrics* implementing this intuition.

4 Permutation complexity: Tight metrics

A complexity metric $C(\cdot)$ is a function from weak factorizations to non-negative reals.

Tight/Semi-tight metrics A complexity metric $C(\cdot)$ is **tight** for a non-primal permutation π iff for every two weak factorizations $\tau_x \neq \tau_y$ of π holds: if τ_x factorizes into τ_y then $C(\tau_x) > C(\tau_y)$. A **semi-tight** metric fulfills the weaker requirement $C(\tau_x) \geq C(\tau_y)$ for all cases except when τ_x is the flattest weak factorization (single node) and $C(\tau_y)$ is a PET, where it strictly requires $C(\tau_x) > C(\tau_y)$.

A metric $C(\cdot)$ is tight iff it is tight for all π. It is semi-tight if it is semi-tight for at least one π and tight otherwise.

Let wf be a weak factorization and let $\mathcal{O}_{wf}$ be the multi-set of (non-leaf) node operators (permutations). We now narrow our attention to functions $F(\cdot)$ over the multi-set $\mathcal{O}_{wf}$, i.e., $C(wf) = F(\mathcal{O}_{wf})$. This means that we are disregarding the bracketing structure of wf. In Section 5 we incorporate bracketings by extending this framework to forests.

What should metric $F(\mathcal{O}_{wf})$ fulfill to be tight? We will parameterize $F(\cdot)$ with a ground metric $C_o(\cdot)$ over node operators, i.e., $F_{C_o}(\cdot)$. The idea here is that higher-order metrics over PETs can better delegate local operator complexity to a dedicated ground metric defined directly over operators, particularly primal permutations.

An operator complexity function $C_o(op)$ monotone non-decreasing[5] in operator length $|op|$ implements the idea that longer primal permutations are more complex,[6] cf. (Brignall, 2010) Theorem 2.2.:

> *Every primal permutation of length $n \geq 2$ contains another primal permutation of length $(n-1)$ or $(n-2)$ (Schmerl and Trotter, 1993).*

For example, $\langle 2,4,1,5,3\rangle$ contains $\langle 2,4,1,3\rangle$, and the latter contains $\langle 2,1\rangle$.[7]

Now we look at functions $F_{C_o}(\cdot)$ that are *monotone increasing* in the arithmetic average of $C_o(\cdot)$: $\textsc{Avg}_{C_o}(wf) = \frac{1}{|\mathcal{O}_{wf}|} \times \sum_{op \in \mathcal{O}_{wf}} C_o(op)$. For semi-tightness $\textsc{Max}_{C_o}(wf)$ is suitable. The following theorem says that a metric is tight if it assigns lower complexity to a permutation factorizing into a PET with shorter average primal permutation length.

Theorem 1 A metric $C(\cdot)$ is tight (semi-tight) if for some $C_o(\cdot)$, monotone non-decreasing in *operator length*, metric $C(\cdot)$ is monotone increasing (respectively non-decreasing) in $\textsc{Avg}_{C_o}(\cdot)$.

Proof Assume τ_0 factorizes to τ_j in a number of steps $j \geq 1$. In every step τ_{i-1} to τ_i, one operator op in τ_{i-1} of length $|op| > 2$ factorizes into $\sigma[op_1, \cdots, op_m]$, where $m \geq 2$. By the nature of factorization, $|\sigma| = m$ and $\sum_{i=1}^{m} |op_i| = |op|$. Let $\mathcal{O}^-$ stand for multi-set $\mathcal{O}$ excluding op. The average length of operators in τ_i is $\frac{1}{|\mathcal{O}_{\tau_i}|} \times (\sum_{p \in \mathcal{O}_{\tau_i}} |p|)$; it can be rewritten into $\frac{1}{m+|\mathcal{O}_{\tau_{i-1}}|} \times (m + |op| + \sum_{p \in \mathcal{O}^-_{\tau_{i-1}}} |p|)$ and again into $\frac{1}{m+|\mathcal{O}_{\tau_{i-1}}|} \times (m + \sum_{p \in \mathcal{O}_{\tau_{i-1}}} |p|)$. The desired inequality $\frac{m+\sum_{p \in \mathcal{O}_{\tau_{i-1}}} |p|}{m+|\mathcal{O}_{\tau_{i-1}}|} < \frac{\sum_{p \in \mathcal{O}_{\tau_{i-1}}} |p|}{|\mathcal{O}_{\tau_{i-1}}|}$ holds under the condition that $\sum_{p \in \tau_{i-1}} |p| > |\mathcal{O}_{\tau_{i-1}}|$, i.e., the average length of operators in a weak factorization is greater than the number of non-leaf nodes. The latter is a tautology because the branching factor of any node is always two or more. Tightness (semi-tightness) follows if $C(\cdot)$ is monotone increasing (respectively monotone non-decreasing) in $\textsc{Avg}_{C_o}(\cdot)$ when C_o is monotone non-decreasing in operator length. □

In other words, a metric assigning lower complexity to more factorizable permutations (by average operator length) is tight, i.e., *it allows comparing permutations by the smallest complexity assigned to them within this framework.* In Figure 3, a tight metric $C(\cdot)$ assigns $C(3a) > C(3b) > C(x)$ for

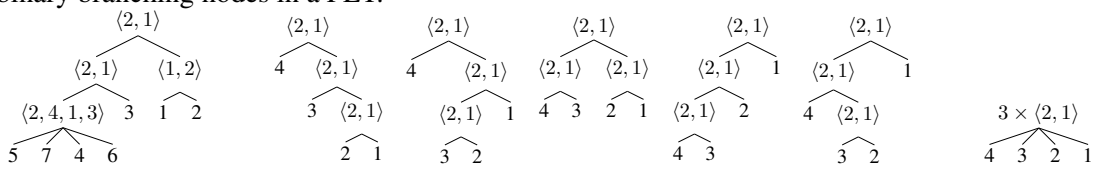
[5] $C_o(op)$ could be *monotone increasing*, but the weaker requirement is sufficient. Practically, we could parameterize $C_o(op)$ in operator-clusters and train it on data.

[6] A further practical requirement is $C_o(op) = 0$ iff $op = \langle 1,2\rangle$. But this is not necessary for tightness.

[7] By definition a primal permutation cannot be a sub-permutation of another primal permutation.

$x \in \{3c, 3d\}$. A semi-tight metric assigns complexity scores such that either $C(3a) \geq C(3b) > C(x)$ or $C(3a) > C(3b) \geq C(x)$ for $x \in \{3c, 3d\}$.

An example tight metric is the number of nodes in a PET; the more nodes the shorter the average length of primal operators. Beside maximum operator length, another semi-tight metric is the number of non-binary branching nodes in a PET.



(a) PET $\langle 5, 7, 4, 6, 3, 1, 2\rangle$. (b) Five different PETs for $\pi = \langle 4, 3, 2, 1\rangle$. (c) Flattened PET

Figure 5: In 5a and 5b PETs for different permutations. In 5c a flattened PET for the five in 5b.

5 From permutation trees to forests

For many computational purposes, a single *canonical PET* is sufficient, cf. (Gildea et al., 2006). A single PET can be computed in linear-time, cf. (Uno and Yagiura, 2000; Zhang and Gildea, 2007). Crucial for efficiency is the *uniqueness* of the sub-permutations for factorizations of arity $\mathbf{a} \geq 4$ (see Albert and Atkinson's result – Section 3), i.e., there is a single choice for a set of *split points* between adjacent sub-permutations. For arity $\mathbf{a} = 2$, there are at most $(n-1)$ choices for a single split point, if $|\pi| = n$. This is also crucial for our Permutation Forest algorithm defined next.

Some permutations factorize into multiple alternative PETs (see Figure 5b). The alternative PETs of π can be packed into an $O(n^2)$ permutation forest (PEF).

Flattened PET In a PET, chains of binary operators, either all inverted or all monotone, can be flattened leading to a *special kind* of packed representation. For example, the monotone operators in the PETs in Figures 3c and 3d can be flattened into the representation in Figure 3b. Therefore, we distinguish this from regular factorization by writing the encapsulated chain explicitly as in $3 \times \langle 2, 1\rangle$ in Figure 5c. This notation means that all binarizations of this order are allowed under that node. Various metrics defined in the next section are computed without the need to unpack this representation (e.g., the number of possible binarizations), which leads to algorithms over PEFs in $O(n)$. In the sequel we refer to function FLAT(PET) which "flattens" PET in this fashion.

A **permutation forest** (akin to a parse forest) $\mathcal{F}$ for π (over $[1..n]$) is a data structure consisting of a subset of $\{[[i, j, \mathcal{I}_i^j]] \mid 0 \leq i \leq j \leq n\}$, where $\mathcal{I}_i^j$ is a (possibly empty) set of *inferences* for π_{i+1}^j. If π_{i+1}^j is a sub-permutation and it has arity $\mathbf{a} \leq (j - (i+1))$, then each inference consists of a $\mathbf{a}$-tuple $[p, l_1, \ldots, l_{\mathbf{a}-1}]$, where the **operator** p is the permutation of the $\mathbf{a}$ sub-permutations ("children" of π_{i+1}^j), and for each $1 \leq x \leq (\mathbf{a}-1)$, l_x is a "split point" which is given by the index of the last integer in the x^{th} sub-permutation in π.

Function PEF$(i, j, \pi, \mathcal{F})$;
`# Args: sub-perm. π over [i..j] and forest F`
Output: Parse-Forest $\mathcal{F}(\pi)$ for π;

begin
1. if $([[i, j, \star]] \in \mathcal{F})$ then return $\mathcal{F}$; `#memoization`
2. $\mathbf{a} := \mathbf{a}(\pi)$;
3. if $\mathbf{a} = 1$ return $\mathcal{F} := \mathcal{F} \cup \{[[i, j, \emptyset]]\}$;
4. For each set of split points $\{l_1, \ldots, l_{\mathbf{a}-1}\}$ do
5. $\quad p := \text{RANKLISTOF}(\pi_1^{l_1}, \pi_{(l_1+1)}^{l_2}, \ldots, \pi_{(l_{\mathbf{a}-1}+1)}^{n})$;
6. $\quad \mathcal{I}_i^j := \mathcal{I}_i^j \cup [p, l_1, \ldots, l_{\mathbf{a}-1}]$;
7. $\quad$ For each $\pi_v \in \{\pi_1^{l_1}, \pi_{(l_1+1)}^{l_2}, \ldots, \pi_{(l_{\mathbf{a}-1}+1)}^{n}\}$ do
8. $\quad\quad \mathcal{F} := \mathcal{F} \cup PermForest(\pi_v)$;
9. $\mathcal{F} := \mathcal{F} \cup \{[[i, j, \mathcal{I}_i^j]]\}$;
10. Return $\mathcal{F}$;
end;

Figure 6: Pseudo-code of permutation-forest factorization algorithm. Function $\mathbf{a}(\pi)$ returns the arity of π. Function RANKLISTOF$(r_1, \ldots, r_m)$ employs *Counting Sort* (Cormen et al., 2001) to sort the sub-permutations $r_1, \ldots, r_m$ as integer ranges in $O(n)$, and returns a permutation p over $[1..m]$ signifying their order. The top-level call is PEF$(\pi, 0, n, \emptyset)$. We will use PEF(π) thereby overloading PEF$(\cdot)$.

Let us exemplify the inferences on $\pi = \langle 4,3,2,1\rangle$ (see Figure 5b) which factorizes into pairs of sub-permutations ($\mathbf{a} = 2$): a split point can be at positions with index $l_1 \in \{1,2,3\}$. Each of these split points (factorizations) of π is represented as an *inference* for the *same root node* which covers the whole of π (placed in entry $[0,4]$); an inference here consists of the permutation $\langle 2,1\rangle$ (swapping the two ranges covered by the children sub-permutations) together with $\mathbf{a}-1$ indexes $l_1,\ldots,l_{\mathbf{a}-1}$ signifying the split points of π into sub-permutations: since $\mathbf{a} = 2$ for π, then a single index $l_1 \in \{1,2,3\}$ is stored with every inference. For the factorization $((4,3),(2,1))$ the index $l_1 = 2$ signifying that the second position is a split point into $\langle 4,3\rangle$ (stored in entry $[0,2]$) and $\langle 2,1\rangle$ (stored in entry $[2,4]$). For the other factorizations of π similar inferences are stored in the permutation forest.

Figure 6 shows a simple top-down factorization algorithm which starts out by computing the arity $\mathbf{a}$ using function $\mathbf{a}(\pi)$. If $\mathbf{a} = 1$, a single leaf node is stored with an empty set of inferences. If $\mathbf{a} > 1$ then the algorithm computes all possible factorizations of π into $\mathbf{a}$ sub-permutations (a sequence of $\mathbf{a}-1$ split points) and stores their inferences together as $\mathcal{I}_i^J$ associated with a node in entry $[[i,j,\mathcal{I}_i^j]]$. Subsequently, the algorithm applies recursively to each sub-permutation.

The Albert and Atkinson uniqueness results for $\mathbf{a} \geq 4$ implies that the number of sets of split points is exactly one. For $\mathbf{a} = 2$ there are at most $n-1$ such sets. This means that line 4 in Figure 6 is at most linear in n. Similarly for $\mathbf{a} \geq 4$ line 7 does at most $(n-1)$ recursive calls, and for $\mathbf{a} = 2$ only two. In total, this algorithm has time complexity $O(n^3)$.

6 Evaluation metrics by factorization

So far we presented the Permutation Complexity class of metrics and defined tightness. In this section we present example tight and semi-tight metrics.

The (semi-)tight metrics presented next are linear-time in permutation length. Each of these metrics concentrates on one aspect of PETs/PEFs: factorization extent ($|\textsc{Pet}|$), bracketing freedom ($\#\textsc{Pets}$), and maximum arity ($\textsc{Max}_{|\mathbf{Op}|}$). These example metrics are summarized in Figure 7.

$$
\begin{aligned}
|\textsc{Pet}|(\pi) &= \frac{\textsc{Count}_{node}(PET(\pi))-1}{n-2} \\
\#\textsc{Pets}(\pi) &= \frac{\textsc{Count}_{pet}(\textsc{Pef}(\pi))-1}{\textsc{Count}_{pet}(\textsc{Pef}(\textsc{Id}^n))-1} \\
\textsc{Max}_{|\mathrm{Op}|}(\pi) &= 1 - \frac{MaxOp(PET(\pi))-2}{n-2}
\end{aligned}
$$

Figure 7: Summary of metrics: $\textsc{Count}_{node}$ is number of nodes in $PET(\pi)$; $MaxOp(PET)$ is maximum operator length in PET; $\textsc{Count}_{pet}(PEF)$ returns count of PETs in PEF.

$|\textbf{PET}|(\pi)$ is the ratio of number of nodes in a PET of π to the number of nodes in PET for $\textsc{Id}^n$. This is a *tight metric cf. Section 4.*

$\#\textbf{PETs}(\pi)$ is the ratio of number of different PETs that π factorizes into to this number for a fully monotone permutation. This metric is semi-tight: consider a flattened PET together with a complexity function based on average operator length – taking into account that flattened nodes receive operator length expressed as monotone decreasing in the Catalan number.

$\textbf{MAX}_{|\mathbf{Op}|}(\pi)$ is one minus the normalized maximum operator length in a PET of π (normalized by the range of lengths, i.e., $[2..n]$).

Having defined tight and semi-tight metrics, next we will evaluate these metrics against a gold standard: human judgements in MT.

7 Experimental setting

Data We use human rankings of translations from WMT13 (Bojar et al., 2013) for ten language pairs with a diverse set of MT systems.

Meta-evaluation We conduct **system level** meta-evaluation by following the method used in (Macháček and Bojar, 2013). All MT systems were first ranked by the ratio of the times they were judged to be better than some other system. All the metrics that we tested compute system level scores for the same systems and then we rank systems by that score (per each metric). The rankings that are

		English-Czech	English-Russian	English-French	English-Spanish	English-German
baselines	HAMMING	0.868 ± 0.033	0.511 ± 0.056	0.911 ± 0.016	0.806 ± 0.056	0.851 ± 0.024
	KENDALL	0.849 ± 0.03	0.511 ± 0.039	0.907 ± 0.014	0.844 ± 0.076	0.918 ± 0.019
	SPEARMAN	0.852 ± 0.029	0.508 ± 0.041	0.907 ± 0.014	0.848 ± 0.074	0.915 ± 0.019
	FUZZY	0.854 ± 0.03	0.498 ± 0.044	0.92 ± 0.014	0.818 ± 0.058	0.897 ± 0.018
	ULAM	0.851 ± 0.029	0.507 ± 0.041	0.914 ± 0.014	0.844 ± 0.07	0.908 ± 0.022
tight	|PET|	0.853 ± 0.029	0.515 ± 0.042	0.907 ± 0.013	0.866 ± 0.074	0.923 ± 0.018
semi	#PETS	0.879 ± 0.053	0.538 ± 0.103	0.904 ± 0.016	0.797 ± 0.052	0.819 ± 0.03
	$\text{MAX}_{\|\text{Op}\|}$	0.849 ± 0.029	0.513 ± 0.043	0.907 ± 0.013	0.864 ± 0.074	0.924 ± 0.018
	BLEU	0.895 ± 0.028	0.574 ± 0.057	0.897 ± 0.034	0.759 ± 0.078	0.786 ± 0.034

Table 1: Correlation with human judgement out of English.

		Czech-English	Russian-English	French-English	Spanish-English	German-English
baselines	HAMMING	0.878 ± 0.028	0.761 ± 0.035	0.984 ± 0.012	0.88 ± 0.033	0.851 ± 0.021
	KENDALL	0.887 ± 0.026	0.831 ± 0.021	0.969 ± 0.012	0.831 ± 0.064	0.905 ± 0.016
	SPEARMAN	0.881 ± 0.025	0.831 ± 0.02	0.967 ± 0.014	0.826 ± 0.066	0.905 ± 0.017
	FUZZY	0.931 ± 0.016	0.81 ± 0.023	0.977 ± 0.009	0.889 ± 0.029	0.894 ± 0.015
	ULAM	0.909 ± 0.026	0.83 ± 0.021	0.974 ± 0.009	0.86 ± 0.054	0.895 ± 0.015
tight	|PET|	0.895 ± 0.026	0.839 ± 0.021	0.965 ± 0.012	0.818 ± 0.064	0.918 ± 0.014
semi	#PETS	0.878 ± 0.034	0.698 ± 0.038	0.959 ± 0.021	0.883 ± 0.042	0.786 ± 0.039
	$\text{MAX}_{\|\text{Op}\|}$	0.895 ± 0.025	0.838 ± 0.021	0.966 ± 0.012	0.819 ± 0.064	0.921 ± 0.014
	BLEU	0.936 ± 0.036	0.651 ± 0.041	0.993 ± 0.014	0.879 ± 0.051	0.902 ± 0.017

Table 2: Correlation with human judgement into English.

produced by all metrics are compared with human judgment using Spearman rank correlation coefficient. When there are no ties, Spearman correlation can be expressed by $\rho = 1 - \frac{6 \sum d_i^2}{n(n^2-1)}$, where $d_i = y_i - x_i$ represents a distance in ranks given by humans and the metric for system i.

Statistical significance We use bootstrap re-sampling with 1000 samples for computing statistical significance. We apply the t-test and we consider a difference significant if $p < 0.05$.

		HAMMING	KENDALL	SPEARMAN	FUZZY	ULAM
Tight	|PET|	**5/4/1**	**7/2/1**	**6/2/2**	**5/4/1**	**5/4/1**
S-tight	$\text{MAX}_{\|\text{Op}\|}$	**5/4/1**	**5/2/3**	**6/2/2**	5/5/0	**5/4/1**
S-tight	#PETS	2/6/2	3/7/0	3/7/0	2/8/0	3/7/0

Table 3: Pairs-wise comparison over 10 language pairs. In the triple $N/B/D$: N is number of language pairs where the new metric significantly outperforms the baseline, B is baseline outperforms new metric and D is the number of language pairs where the difference is insignificant (draw). Bold show the cases where $N > B$.

Evaluating reordering All the tested metrics are defined on the sentence level. Since words in the reference or system translations might not be aligned, we introduce a brevity penalty for the ordering component as in (Isozaki et al., 2010).[8] After scaling sentence-level reordering score $ordering(\pi)$ by a brevity-penalty $BP(|\pi|, |ref|)$, we interpolate the result with a reordering-free (bag-of-words) lexical score $F1(ref, sys)$, i.e.,[9] $SenScore(ref, sys) = \alpha \times F1(ref, sys) + (1 - \alpha) \times BP(|\pi|, |ref|) \times ordering(\pi)$, where π is the permutation represent-

[8]This is the same as in BLEU with the small difference that instead of taking the length of system and reference translation as its parameters, it takes the length of the system permutation and the length of the reference.

[9]$F1 = 2 \times \frac{precision \times recall}{precision + recall}$, where $precision(ref, sys) = \frac{|ref \cap sys|}{|sys|}$ and $recall(ref, sys) = \frac{|ref \cap sys|}{|ref|}$, assuming each of ref and sys is represented as a bag of words.

ing the word alignment between sys and ref. The interpolation parameter was fixed $\alpha = 0.5$, weighing both lexical and reordering metrics equally, to avoid introducing preference for one over the other, but in principle this could be tuned on human rankings.

We score a system $\mathcal{S}$ by aggregating $SenScore$ weighted by reference length over reference-system pairs in the system's corpus $\mathcal{C}_\mathcal{S}$ and normalize: $Score(\mathcal{S}) = \frac{\sum_{(ref,sys)\in\mathcal{C}_\mathcal{S}} |ref| \times SenScore(ref,sys)}{\sum_{(ref,sys)\in\mathcal{C}_\mathcal{S}} |ref|}$.

Word alignments We align system and reference translations directly using the METEOR aligner (Denkowski and Lavie, 2011), which implements beam search over all possible monolingual alignments that could be built with exact, stem, WordNet and paraphrase match, where each matching mode is weighted depending on language pair.[10]

All metrics in our experiments are interpolated in the same manner with lexical component and brevity penalty, and are fed with the same input permutations.

Results The scores for translation into-English are in Table 2. Table 1 shows the results for the out-of-English direction. We also include BLEU-Moses straight from WMT13 tables for an impression regarding a known full metric. The present tight/semi-tight metrics outperform the baselines on six language pairs (English into Czech/ Russian/ Spanish/ German, and out of Russian and German). But the baselines prevail on four (English into French, and out of Czech, French and Spanish). We hypothesize that English-French shows local reordering where hierarchical factorization has small effect. The results for French- and Spanish-English might be explained similarly. For English-Russian and English-Czech, #PETs (bracketing freedom) is superior, likely because Russian and Czech allow freer order than English which is difficult for MT systems to capture. English-Russian shows low correlations for all metrics (including BLEU), suggesting that either all systems participating are judged of lower quality, or that human judgements are less consistent. For Czech-English, FUZZY, which outperforms all metrics, concentrates on monotone patterns suggesting that Czech-English MT systems in WMT13 differ mainly in how well they obtain correct phrases/blocks in their translations rather than long-distance ordering.

Comparison between ten metrics over ten language pairs is difficult. Hence, we present a pair-wise comparison between the metrics. Table 3 shows for each new metric $\mathcal{N}$ and baseline $\mathcal{B}$ a ratio $N/B/D$ where N is the number of language-pairs where statistically significant improvement by $\mathcal{N}$ over $\mathcal{B}$ is found, B is the reverse situation and D is the number of draws (insignificant difference).

Table 3 shows clearly that the tight metric |PET| performs more often than not better than each of the baselines. Semi-tight metric $\text{MAX}_{|\text{Op}|}$ concerns factorizability and performs as well as FUZZY outperforming the other baselines. Semi-tight metric #PETs concerns bracketing freedom and performs worse than many baselines, suggesting that for most language pairs bracketing freedom, which does not always favor more factorization, is not sufficient. Furthermore, tight and semi-tight metrics |PET| and $\text{MAX}_{|\text{Op}|}$ outperform the *not* semi-tight metrics suggesting that the improvement comes from (semi-)tightness rather than arbitrary functions over trees.

Our results exemplify that factorizing word order mismatch might have higher chance of correlating with human evaluation than the baselines. The tight and semi-tight metrics tested here are simple instantiations that illustrate the general class. More effective variants do more justice to the complexity of primal permutations. Furthermore, different metrics cover different dimensions of complexity. The results show that the importance of a dimension depends on the language pair.

8 Conclusions

The factorized representations of permutations as PETs and PEFs bring together two ingredients (1) grouping words into blocks, and (2) factorization into primal permutations. In this paper we propose a class of metrics, Permutation Complexity, define and show tightness for a sub-class, extend PETs to PEFs and explore example (semi-)tight evaluation metrics exploiting both the hierarchical and primality dimensions. Experiments with WMT13 data show that tight or semi-tight metrics compare favorably

[10]We also make METEOR minimize the number of unaligned words using "-t maxcov".

to the baselines in correlation with human evaluation. Our results can be seen as novel evidence suggesting that tightness might constitute a guiding principle for word order evaluation. The metrics presented in this work only exemplify the range of possible metrics based on the same intuition. In future work we aim at further ordering of the space of metrics, exploring a variety of new complexity metrics, and testing their value on various (evaluation) tasks.

Acknowledgments

This work is supported by STW grant nr. 12271 and NWO VICI grant nr. 277-89-002.

References

Michael H. Albert and Mike D. Atkinson. 2005. Simple permutations and pattern restricted permutations. *Discrete Mathematics*, 300(1-3):1–15.

Alexandra Birch and Miles Osborne. 2010. LRscore for Evaluating Lexical and Reordering Quality in MT. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 327–332, Uppsala, Sweden, July. Association for Computational Linguistics.

Alexandra Birch and Miles Osborne. 2011. Reordering Metrics for MT. In *Proceedings of the Association for Computational Linguistics*, Portland, Oregon, USA. Association for Computational Linguistics.

Alexandra Birch, Miles Osborne, and Philipp Koehn. 2008. Predicting success in machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 745–754, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alexandra Birch, Miles Osborne, and Phil Blunsom. 2010. Metrics for MT evaluation: evaluating reordering. *Machine Translation*, pages 1–12.

Arianna Bisazza and Marcello Federico. 2013. Dynamically shaping the reordering search space of phrase-based statistical machine translation. *TACL*, 1:327–340.

Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria, August. Association for Computational Linguistics.

Fabienne Braune, Anita Gojun, and Alexander Fraser. 2012. Long-distance reordering during search for hierarchical phrase-based SMT. In *Proceedings of the European Association for Machine Translation (EAMT12)*, pages 177–184, Trento, Italy.

Robert Brignall. 2010. A survey of simple permutations. In Steve Linton, Nik Ruškuc, and Vincent Vatter, editors, *Permutation Patterns*, volume 376 of *London Math. Soc. Lecture Note Ser.*, pages 41–65. Cambridge Univ. Press, Cambridge.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 2(33):201–228.

Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. 2001. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition.

John DeNero and Jakob Uszkoreit. 2011. Inducing sentence structure from parallel corpora for reordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 193–203, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*.

Revital Eres, Gad M. Landau, and Laxmi Parida. 2004. Permutation pattern discovery in biosequences. *Journal of Computational Biology*, 11(6):1050–1060.

Yang Gao, Philipp Koehn, and Alexandra Birch. 2011. Soft dependency constraints for reordering in hierarchical phrase-based translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 857–868, Stroudsburg, PA, USA. Association for Computational Linguistics.

Daniel Gildea, Giorgio Satta, and Hao Zhang. 2006. Factoring Synchronous Grammars by Sorting. In *ACL*.

Keith Hall, Ryan McDonald, and Slav Petrov. 2011. Training structured prediction models with extrinsic loss functions. In *Domain Adaptation Workshop at NIPS*, October.

Teresa Herrmann, Jochen Weiner, Jan Niehues, and Alex Waibel. 2011. Analyzing the potential of source sentence reordering in statistical machine translation. In *Proceedings of International Workshop on Spoken Language Translation (IWSLT)*.

Liang Huang, Hao Zhang, Daniel Gildea, and Kevin Knight. 2009. Binarization of synchronous context-free grammars. *Computational Linguistics*, 35(4):559–595.

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 944–952, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jason Katz-Brown, Slav Petrov, Ryan McDonald, Franz Och, David Talbot, Hiroshi Ichikawa, Masakazu Seno, and Hideto Kazawa. 2011. Training a parser for machine translation reordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 183–192, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mirella Lapata. 2006. Automatic Evaluation of Information Ordering: Kendall's Tau. *Computational Linguistics*, 32(4):471–484.

Matouš Macháček and Ondřej Bojar. 2013. Results of the WMT13 Metrics Shared Task. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 45–51, Sofia, Bulgaria, August. Association for Computational Linguistics.

Graham Neubig, Taro Watanabe, and Shinsuke Mori. 2012. Inducing a discriminative parser to optimize machine translation reordering. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 843–853, Stroudsburg, PA, USA. Association for Computational Linguistics.

James H. Schmerl and William T. Trotter. 1993. Critically indecomposable partially ordered sets, graphs, tournaments and other binary relational structures. *Discrete Mathematics*, 113(1-3):191–205.

Miloš Stanojević and Khalil Sima'an. 2014a. Evaluating Word Order Recursively over Permutation-Forests. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 138–147, Doha, Qatar, October. Association for Computational Linguistics.

Miloš Stanojević and Khalil Sima'an. 2014b. Fitting Sentence Level Translation Evaluation with Many Dense Features. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 202–206, Doha, Qatar, October. Association for Computational Linguistics.

David Talbot, Hideto Kazawa, Hiroshi Ichikawa, Jason Katz-Brown, Masakazu Seno, and Franz Och. 2011. A Lightweight Evaluation Framework for Machine Translation Reordering. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 12–21, Edinburgh, Scotland, July. Association for Computational Linguistics.

Takeaki Uno and Mutsunori Yagiura. 2000. Fast algorithms to enumerate all common intervals of two permutations. *Algorithmica*, 26(2):290–309.

Wei Wang, Kevin Knight, and Daniel Marcu. 2007. Binarizing syntax trees to improve syntax-based machine translation accuracy. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 746–754.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 3(23):377–403.

Bing Xiang, Niyu Ge, and Abraham Ittycheriah. 2011. Improving reordering for statistical machine translation with smoothed priors and syntactic features. In *Proceedings of the Fifth Workshop on Syntax, Semantics and Structure in Statistical Translation*, SSST-5, pages 61–69, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hao Zhang and Daniel Gildea. 2007. Factorization of Synchronous Context-Free Grammars in Linear Time. In *NAACL Workshop on Syntax and Structure in Statistical Translation (SSST)*, pages 25–32.