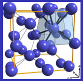
# Latent Reordering Grammar

## Compositional Structure in MT



Khalil Sima'an

Institute for Logic, Language and Computation (ILLC)
University of Amsterdam, The Netherlands

# Reordering Grammar and Compositional Structure

**With Miloš Stanojević**

# Recursive Translation Equivalence (Phrase Pairs)



Translation equivalents in a parallel corpus:

Sentential trans. equiv. are source-target sentence pairs (given)

Atomic translation equivalents = word alignments (induced)

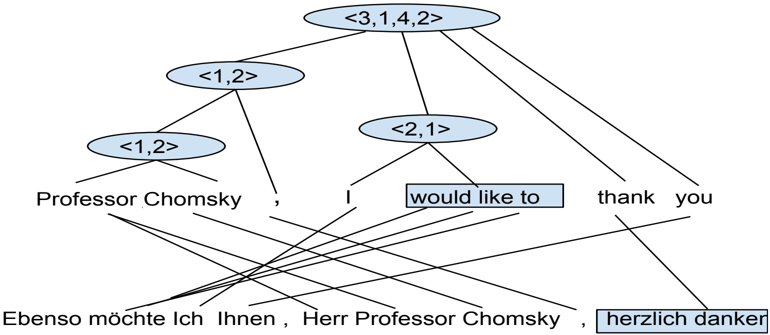
Assumption Translation equivalents limited to phrase pairs

**Seek** a tree structure explaining sentence level from atomic level:

- All phrase pairs covered in a recursive tree structure.
- Tree structure shows subsumption of phrase pairs (composition)
- Tree structure shows recursive reordering (composition)

**Which tree structure?**

# Intuition: Permutation Trees (PETs)




Interpretation root node: Put first child as third, second as first...
Operators are not necessarily binary! (non-ITG).

Factorizing permutations for SCFGs: Permutation Trees (Gildea and Zhang 2006; Zhang et al 2007)
Factorizing word alignments (Sima'an and Maillette DBW 2011)

# Properties: Permutation Trees (PETs)



Formal properties (Albert and Atkinson 2005):

- The operators on the PET are unique and non-decomposable: **Prime Permutations**!
  Example Prime Perms: $\langle 1, 2\rangle$, $\langle 2, 1\rangle$, $\langle 2, 4, 1, 3\rangle$, $\langle 3, 1, 4, 2\rangle$...
- Every permutation decomposes/factorizes into PETs

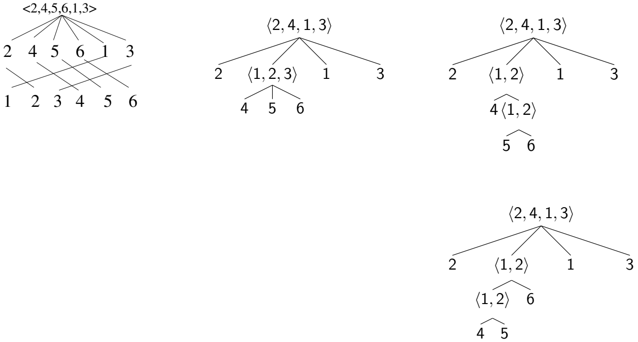
Coverage and composition properties:

- Every phrase pair is covered by a node in a PET!
- Subsumption of phrases $==$ parent-child for nodes.
- Multiple PETs for same permutation (same operators, different binary bracketting)

Hierarchical Alignment Trees (HATs – Sima'an and MdBW 2011) extend PETs and have similar properties.

# Another example: Factorizing permutations

Suppose the alignments are simplified into permutations over minimal translation units:




Multiple permutation Trees (PETs) per word alignment.

# The Hidden Treebank

Word-aligned parallel corpus == Treebank over source sentences:

- PETs obtained from factorizing word alignments.
  Explaining phrase composition recursively
- PETs go beyong ITG (binarizable permutations). Prime Permutations of any arity.
- Hidden treebank Many PETs from a word alignment.
  An ambiguous treebank!
- Unlabeled trees: PET nodes do not have labels.
  Transduction operators on the nodes but no labels.

**What to do with a (Hidden) Treebank?**

# Reminder from treebank parsing



Little reminder from treebank parsing:

- Wall Street Journal treebank for English
- Extract PCFG from treebank (or subtrees)
- Automatically refine treebank labels to fit data

  Label refinement with EM

  (cf. Prescher 2005; Matsuzaki et al 2005; Petrov 2006/7)
- $\Rightarrow$ A PCFG with labels refined to fit data

  Refinement reduced ambiguity and increases accuracy.

  Among the best results in monolingual parsing.

Apply similar approach to word alignments?

# Challenges with PETs Treebank



**PETs Treebank** For every word aligned sentence pair:

1. Write target positions as a permutation of source positions.
2. Factorize permutation into PETs over source sentence.

Manual clustering (Maillette DBW & Sima'an SSST 2013,2014)

Pecularities for applying EM for label refinement:

No labels! Our PETs do not have node labels like NP, VP!
**Solution** Prime Permutations as initial labels.
Refine prime permutations: Reordering labels!!

Hidden! Word alignment defines many PETs, not one!
**Solution** Pack PETs into parse-forest in $O(n^3)$
Induce distribution over PETs!

What to do with Reordering PCFG after learning?

# Possible Uses of Reordering Grammar



- As pre-ordering model
- As reordering model in phrase systems
- As synchronous grammar for MT

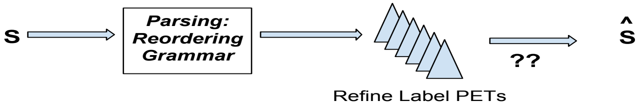
This talk: Preordering only

Related work on inducing preordering all with ITG:

- Tromble and Eisner EMNLP 2009. Learn Kendall tau reordering table and use binary trees.
- DeNero and Uszkoreit EMNLP 2011. Induce unlabeled binary tree (brackets), and separately train a reordering model.
- Neubig et al EMNLP 2012. Induce binary trees with separate reordering as well.

# First Use Case: Preordering s to ŝ




1. Learn (EM) label refined PCFG (Reordering G.)
2. Use Reordering Grammar to parse a source sentence
   Refined node labels correspond to prime perms!
3. Obtain reordered version of source sentence.

Reordering Grammar: Because labels are Prime Permutations

# Some technical difficulties and solutions



**Complexity!** Explosion of number of rules.
Unary rule trick makes this manageable.
Unary trick: only pairs of labels

**Reordering!** We need reordering *not parse trees!*
Given refined PCFG $\mathcal{G}$:

$$\arg\max_{\pi} P(\pi) = \arg\max_{\pi} \sum_{\Delta \in PETs(\pi)} \sum_{d(\Delta) \in \mathcal{G}} \prod_{r \in d(\Delta)} P(r)$$

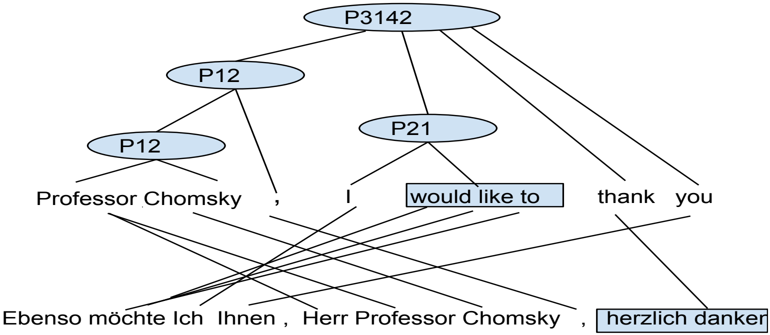
Highest probability permutation is NP-Complete (Sima'an 1996)

Minimum-Bayes Risk Decoding for reordering computed from PCFG expectations over labele refined PETs optimizing Kendall tau

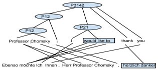
Details to be released soon

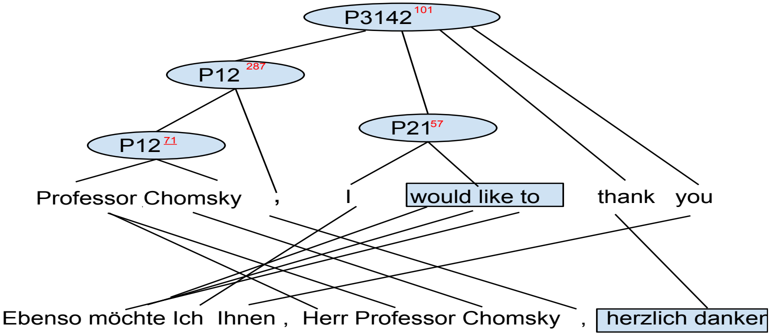
# Initial Labels in Hidden Treebank





# Refined labels after learning






Crucial: Refined labels == Unambiguous for reordering

# Experiments English-Japanese



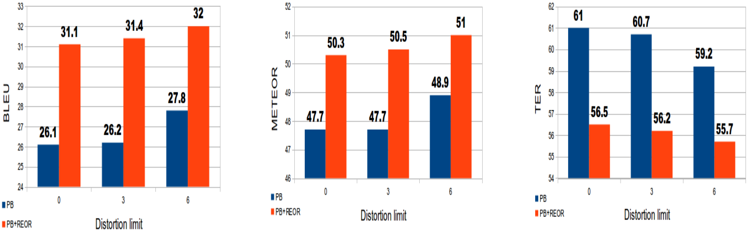
| corpus | #sents | #words source | #words target |
|---|---|---|---|
| train reordering | 786k | 21M | – |
| train translation | 950k | 25M | 30M |
| tune translation | 2k | 55K | 66K |
| test translation | 3k | 78K | 93K |

- English-Japanese - NTCIR-8 Patent Translation (PATMT).
- Standard dev set (NTCIR-7) en test sets (NTCIR-9).
- Reordering Grammar: 10 iterations of EM (2 days).
- Testing on test set 11 hours.

**Back-end Phrase-Based System $\hat{\mathbf{s}} \rightarrow \mathbf{t}$**: 5-gram LM; tuning 3-times with kb-Mira and evaluate with Multeval.

# Preordering vs. distortion limit

Varying distortion limit in back-end system:




Baseline in **blue** and preordering in **red**.

(1) Major improvement by preordering.

(2) Preordering Grammar works well with all distortion limits!

# Preordering with MSD Reordering



Lexicalized (MSD) reordering back-end system + distortion=6:

| **Metric** | **System** | **Avg** | ***p*-value** |
|---|---|---|---|
| BLEU ↑ | PB MSD | 29.6 | - |
| | PB MSD + REOR | 32.4 | 0.00 |
| METEOR ↑ | PB MSD | 50.1 | - |
| | PB MSD + REOR | 51.3 | 0.00 |
| TER ↓ | PB MSD | 58.0 | - |
| | PB MSD + REOR | 55.3 | 0.00 |

MSD improves over distortion model but

Preordering Grammar still gives major improvement.

# Preordering vs Hierarchical Model (Hiero)

Preordering does not have access to target words.

Hence: Preordering cannot solve all reorderings!

But how does Preordering Grammar fair against Hiero?

# Preordering vs Hierarchical Model (Hiero)



Preordering does not have access to target words.

Hence: Preordering cannot solve all reorderings!

But how does Preordering Grammar fair against Hiero?

| **Metric** | **System** | **Avg** | ***p*-value** |
|---|---|---|---|
| BLEU ↑ | Hiero | 32.6 | - |
| | PB MSD + REOR | 32.4 | 0.16 |
| METEOR ↑ | Hiero | 52.1 | - |
| | PB MSD + REOR | 51.3 | 0.00 |
| TER ↓ | Hiero | 54.5 | - |
| | PB MSD + REOR | 55.3 | 0.00 |

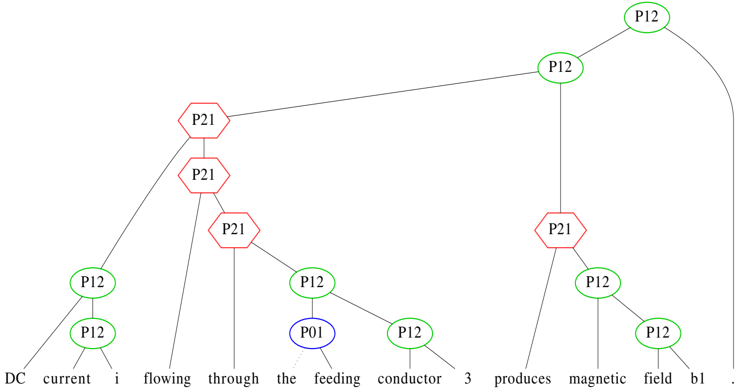
Preordering Grammar insignificantly different from Hiero!

Preordering Grammar only on source side!

- No synchronous grammar: PCFG
- No lexicalized reordering: reordering labels
- No long tables: compositional and learned from data!

# Summary of results and example






- the article “the” does not have an equivalent in Japanese,
- verbs go after their object
- use postpositions instead of prepositions
- prefer grouping certain syntactic units (in this example NPs and VPs)

# Summary of talk

Topic Composition and translation equivalence.

- How to fit monolingual syntax to MT?
- This demands statistical learning on parallel data
- Not a proper fit and not likely to always improve

- Reverse question for MT:

  Which structure underlies data?

- Factorizing word alignments (or learning bilingual trees)
- PETs and Hierarchical Alignment Trees (HATs)
- Reordering Grammar learned by refining permutations
- Gives improved performance for pre-ordering