
Factorization of Multitext Grammars as Multi-dimensional Permutations

Hao Zhang and Daniel Gildea

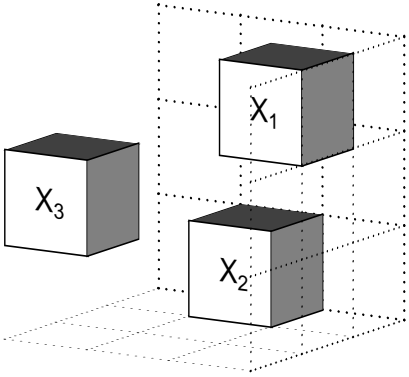
Computer Science Department

University of Rochester

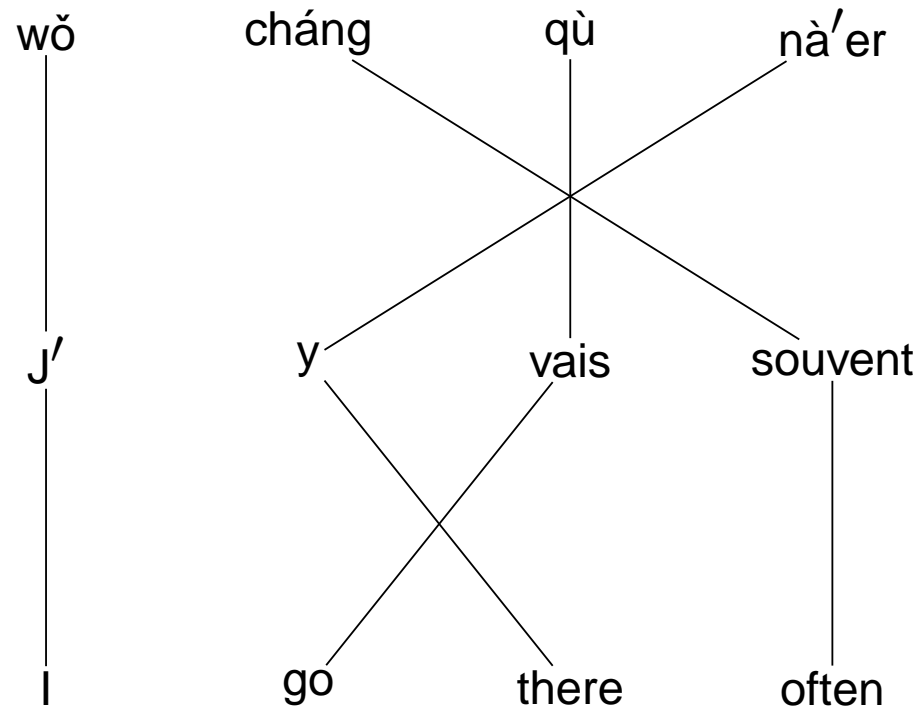
Outline

- From Multitext Grammars (MTG) to Multi-dimensional Permutations
- Factorizing Multi-dimensional Permutations into Trees
- Enumerating Multi-dimensional Permutation Trees
- Asymptotics

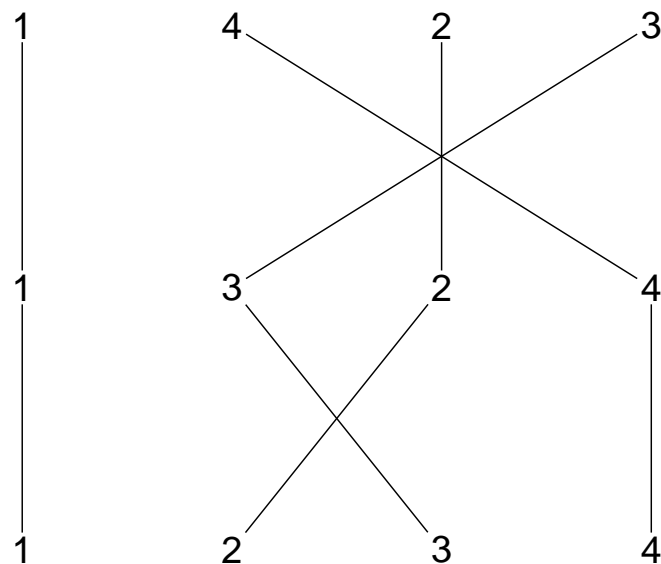
CFG, Synchronous CFG and Multitext Grammars

	CFG	SCFG	3-MTG
terminals	words	word pairs	word triples
nonterminals	phrases	phrase pairs	phrase triples
productions	sequences	<i>permutations</i>	<i>2-permutations</i>
$X \rightarrow$	$X_1 X_2 X_3$	$\begin{bmatrix} & & X_3 \\ X_1 & & \\ & X_2 & \end{bmatrix}$	
parsing	over spans	over cells	over cubes

Multitext Alignment

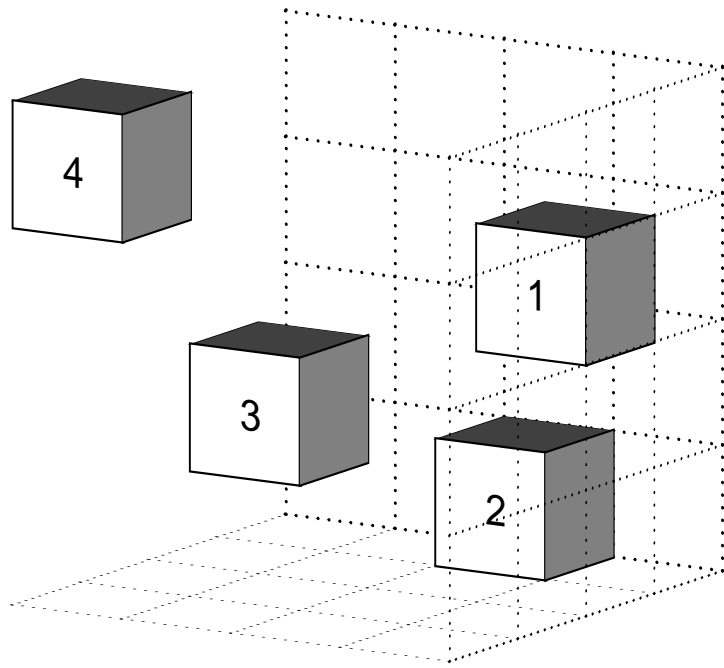


Alignment View of Two-dimensional Permutations

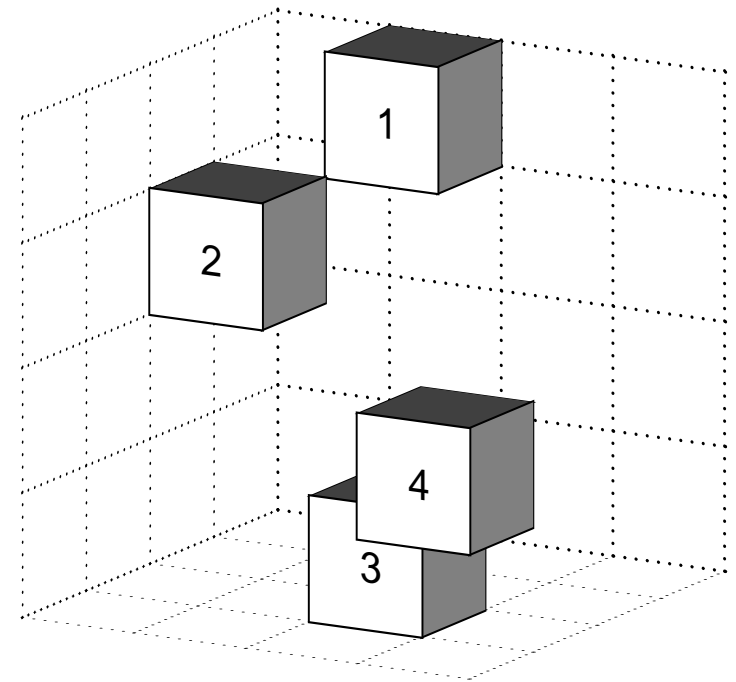


$$\pi = (\text{id}_n = (1, 2, 3, 4), \pi_1 = (1, 3, 2, 4), \pi_2 = (1, 4, 2, 3))$$

3D View of Two-dimensional Permutations

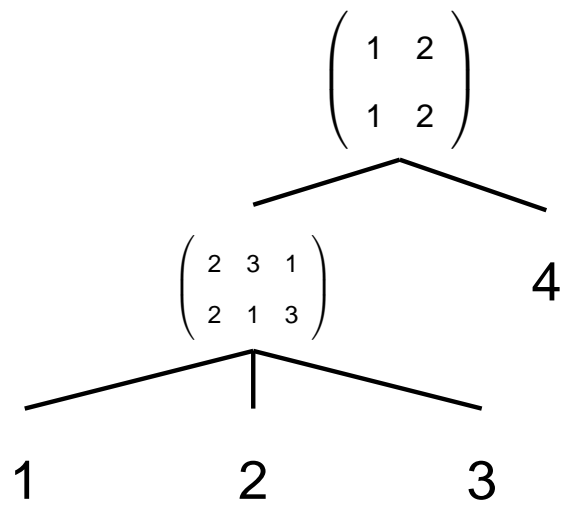


$$\pi = (\text{id}_4, (2, 1, 3, 4), (2, 3, 1, 4))$$

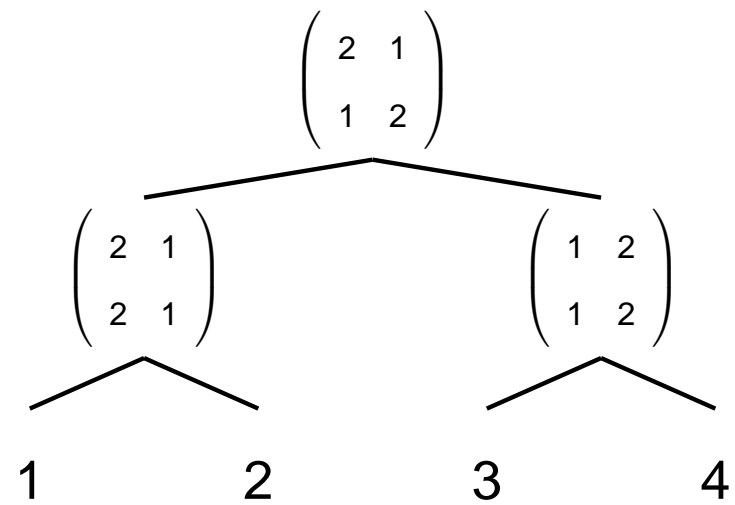


$$\pi' = (\text{id}_4, (2, 1, 3, 4), (3, 4, 2, 1))$$

Two-dimensional Permutation Trees



ternary

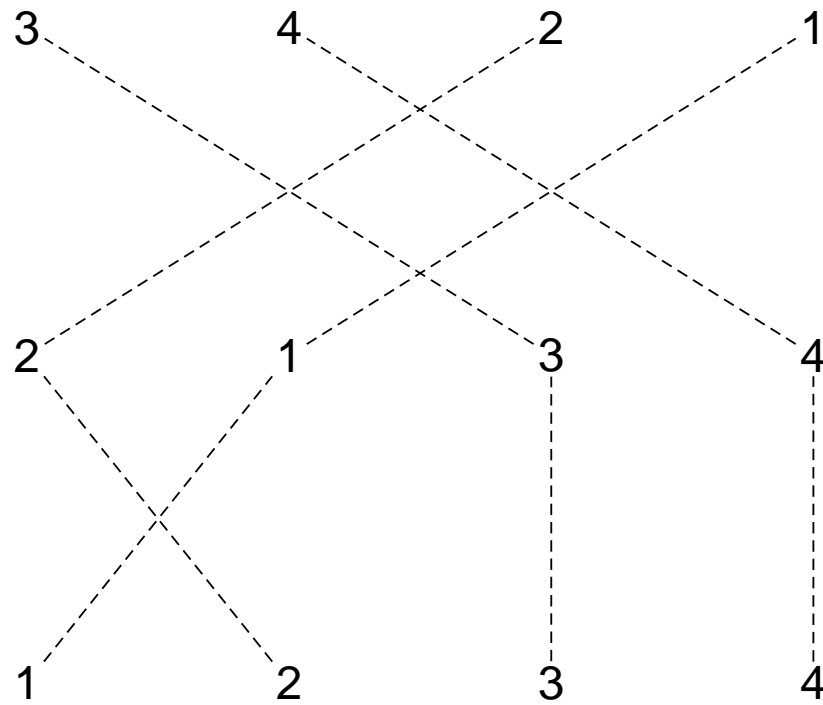


binary

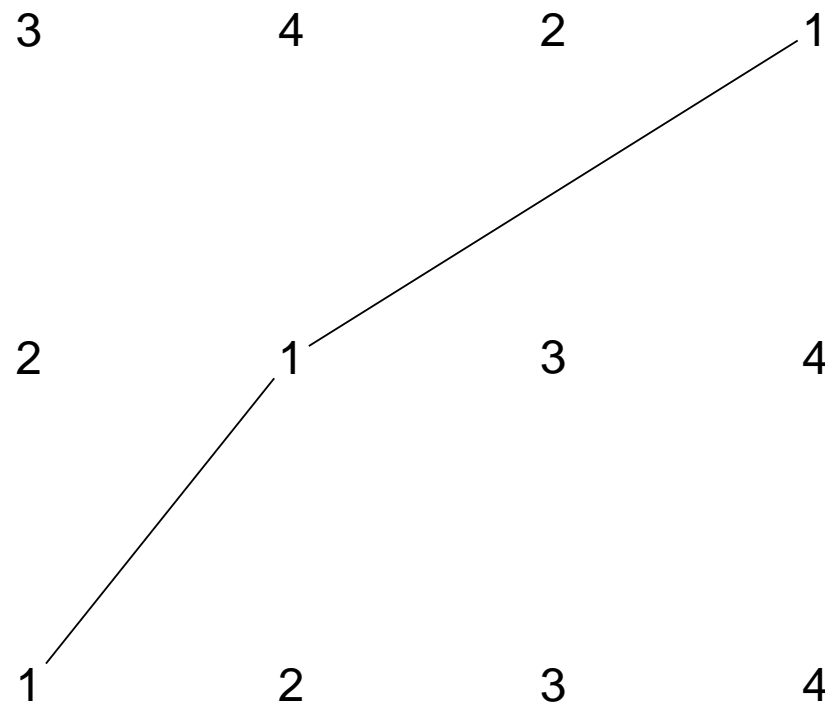
Factorization Problem

- Given a d -permutation, produce a k -ary d -permutation tree with k minimized.
- Basic idea: shift-reduce on the alignment links, make reductions whenever possible, $O(d \cdot n^2)$.
- Optimization: apply Uno&Yagiura (2000) algorithm for eliminating impossible reduction boundaries, $O(d \cdot n)$.
- Independently discovered earlier by Bui-Xuan et.al (2005) using PQ tree as the representation tool. Oh, no...

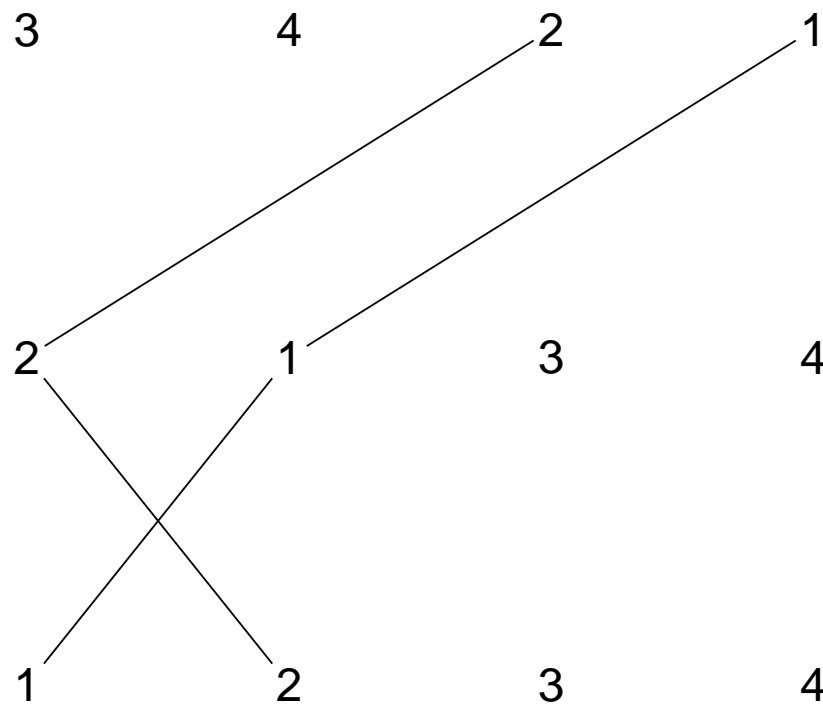
Factorization Algorithm



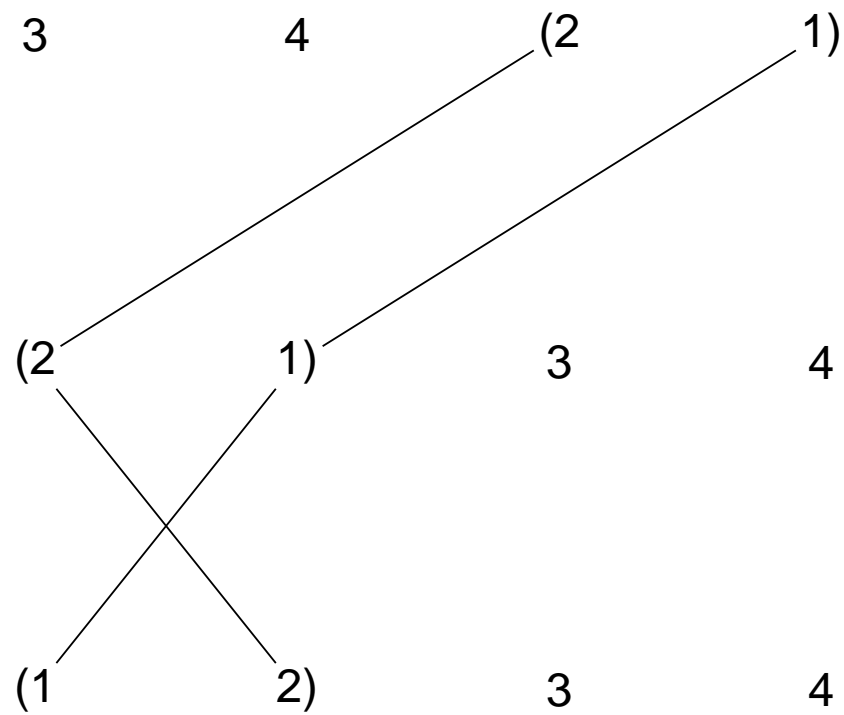
Factorization Algorithm



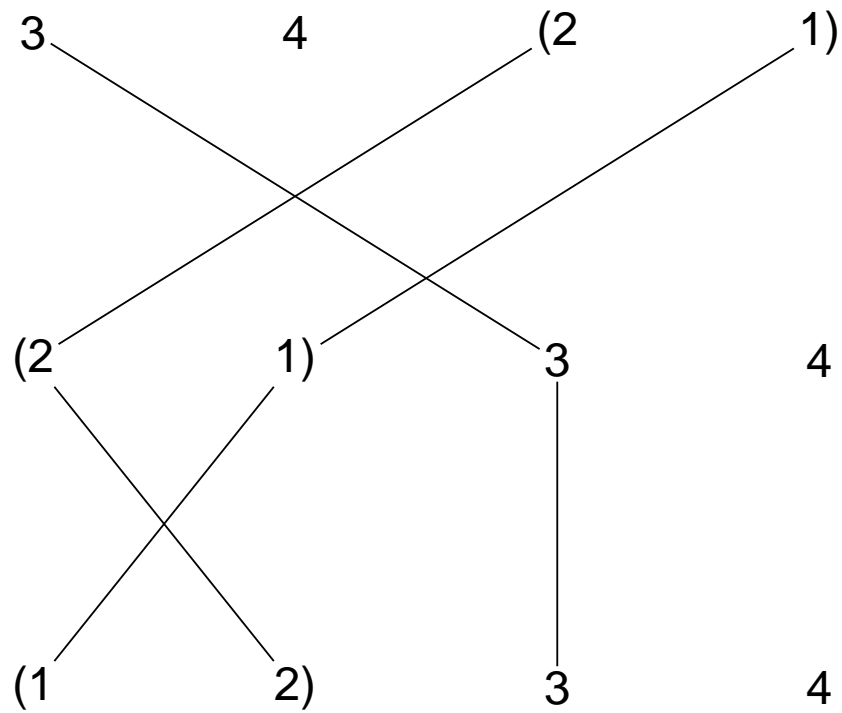
Factorization Algorithm



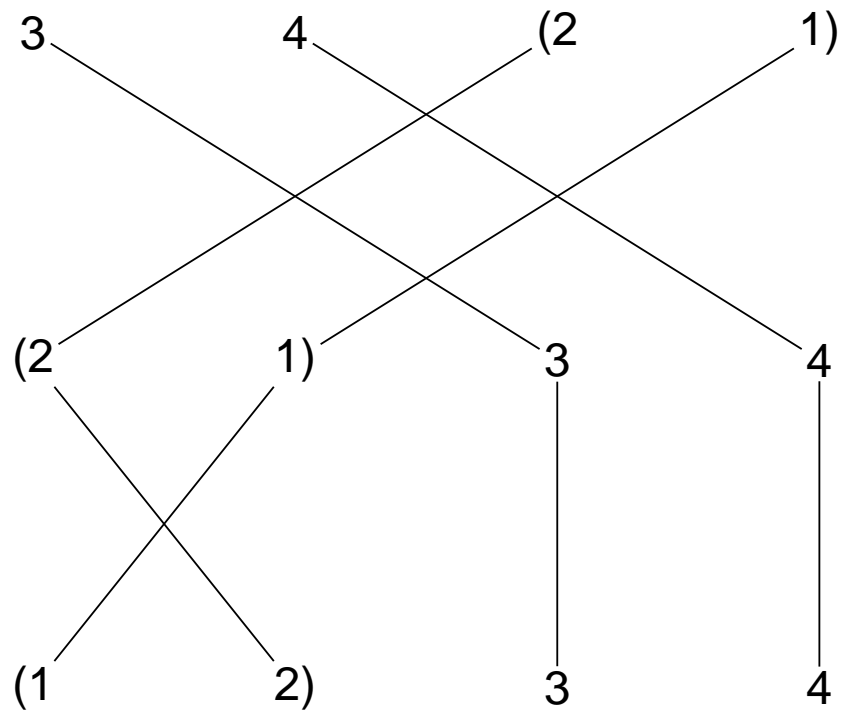
Factorization Algorithm



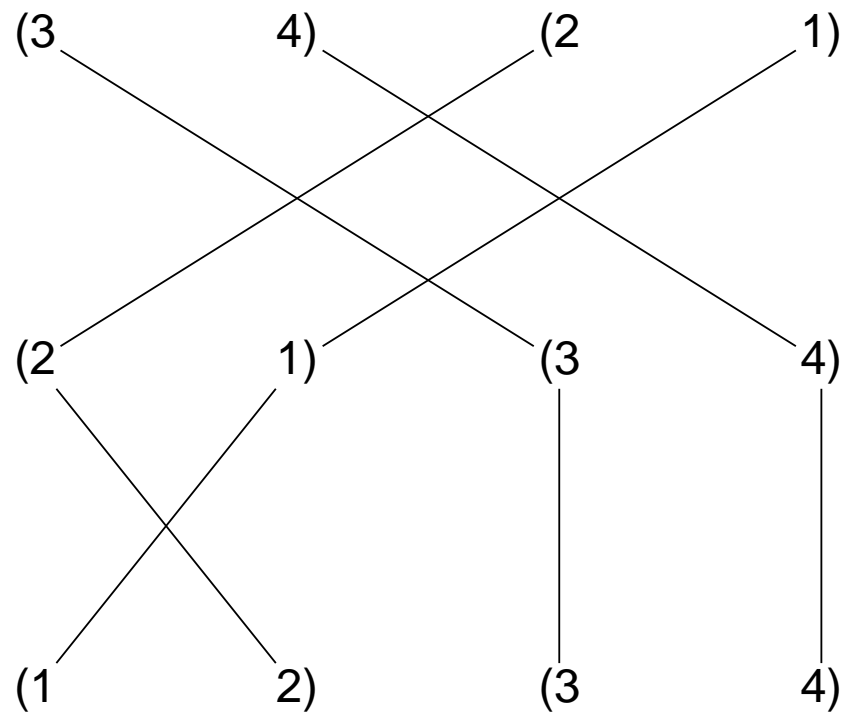
Factorization Algorithm



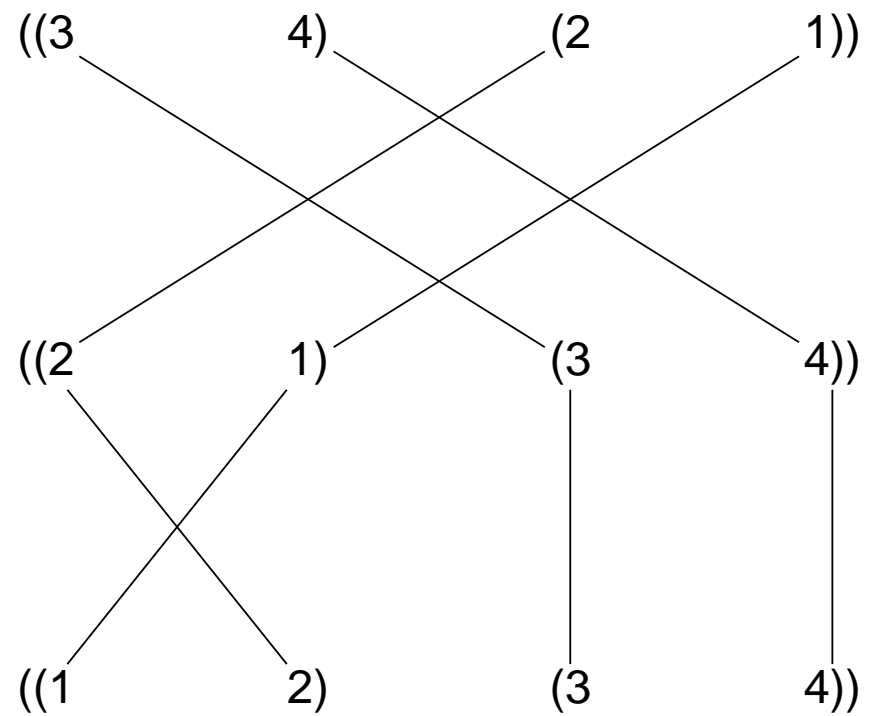
Factorization Algorithm



Factorization Algorithm



Factorization Algorithm



Enumeration of k-arizable d-permutations

- d-permutation trees are unique upon normalization.
- A normalized d-permutation tree has two properties.
 - Minimized: a k-ary d-permutation is used if and only if it is not k – 1 parsable.

$$H_{d,k} = (k!)^d - S_{d,k-1}[k]$$

- Left-most: if a node is binary, its right child must not be binary with the same orientation.

Generating Function for $S_{d,k}$

$$R_{d,k}(x) = x + xR_{d,k}(x) + (2^d - 1)R_{d,k}^2(x) + \sum_{k'=3}^k H_{d,k'} \left(R_{d,k}^{k'}(x) + R_{d,k}^{k'+1}(x) \right)$$

It is algebraic.

Simple d-permutations

k	$H_{2,k}$	
1	1	
2	4	
3	8	$\frac{H_{1,n}}{n!} \rightarrow e^{-2}$
4	172	$\frac{H_{d,n}}{(n!)^d} \rightarrow 1, (d \geq 2)$
5	5204	$H_d, (d \geq 2)$ has not been discovered before!
6	222716	
7	12509188	
8	889421564	

Growth Rate of $S_{d,k}$

$$G_{d,k} = \lim_{n \rightarrow \infty} \frac{S_{d,k}[n]}{S_{d,k}[n-1]}$$

	$G_{2,k}$		$G_{2,k}$
2	13.93	12	57.60
4	22.08	14	69.17
6	29.97	16	82.01
8	38.19	18	96.10
10	47.31	20	111.41

$$G_{d,k} = \left(\frac{k}{e}\right)^d + O(k^{d-1} \cdot \log k)$$

Conclusions and Questions

- If all d-permutations are equally likely, with probability nearly one a d-permutation is not factorizable.
- Can natural language multitext grammars use relatively small k to cover real multitexts? Seems we do need gaps...
- If we introduce a fixed number of gaps, how much more coverage can we achieve, theoretically?
- In terms of introducing gaps, there are still many open questions even for two languages. For example, can we characterize the difference between STAG and SCFG from the perspective of alignment (permutation) coverage?