

# Synchronous Binarization for Machine Translation

Hao Zhang (University of Rochester), Liang Huang (University of Pennsylvania),  
Daniel Gildea (University of Rochester), Kevin Knight (University of Southern California)

## Introduction

- Synchronous grammars are the framework of syntax-based machine translation.
- Both alignment and decoding are parsing problems in this framework.
- Factorization of synchronous grammars can drastically reduce parsing complexity.
- Binarization of Synchronous CFG (SCFG) is the focus of this work.

## Synchronous CFG vs. CFG

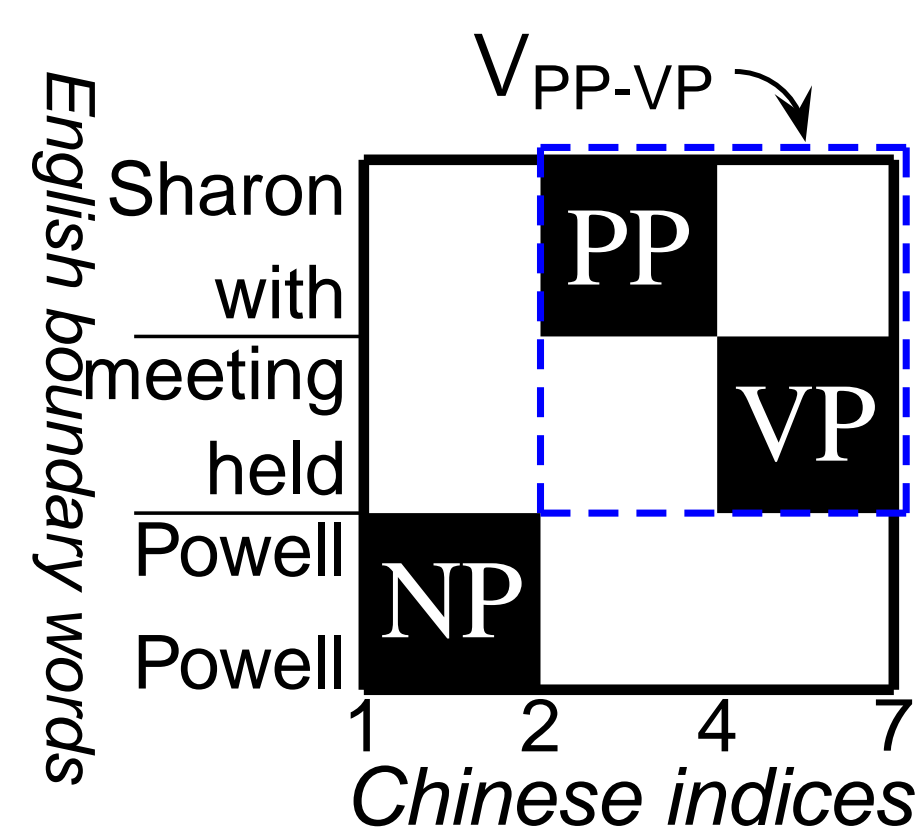
	CFG	SCFG
terminals	words	word pairs
nonterminals	phrases	phrase pairs
productions	sequences	permutations
	$X \rightarrow X_1 X_2 X_3$	$X \rightarrow \begin{bmatrix} & & X_3 \\ X_1 & & \\ & X_2 & \end{bmatrix}$
parsing	over spans	over cells

## Parsing Complexity of n-ary SCFG

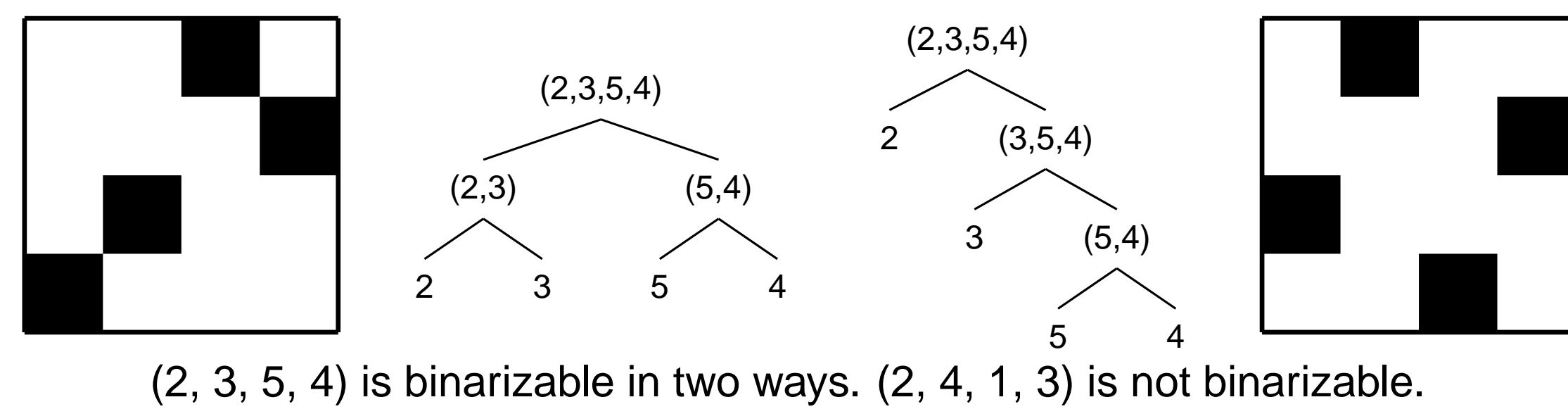
Assume the sentence length (on either side) is  $O(N)$ .

- Naïve parsing is  $O(N^{2n+2})$ , by enumerating at most  $2(n+1)$  split points into the two sentences.
- $O(N^{n+1+3}) = O(N^{n+4})$  is also doable, if we go from left to right on one side and pick up one pebble a time, allowing discontinuous spans on the other side.
- But even if the strategy allows discontinuous spans on both sides, it is  $\Omega(N^{c\sqrt{n}})$ , (Satta and Peserico, 2005), because there exist hard-for-parsing permutations out of  $n!$ .
- When  $n = 2$ , the parsing complexity  $O(N^6)$  is the theoretical lower bound for synchronous parsing using tabular methods.

## Synchronous Binarization Maintains Continuity

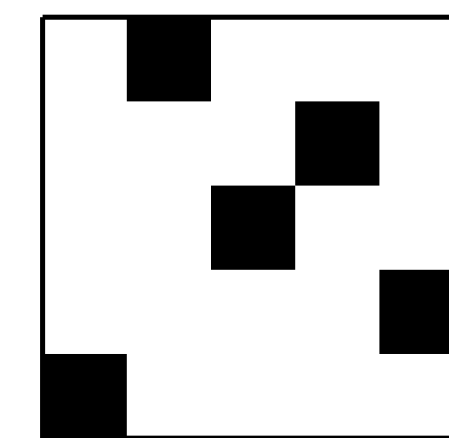


## Permutation Structures

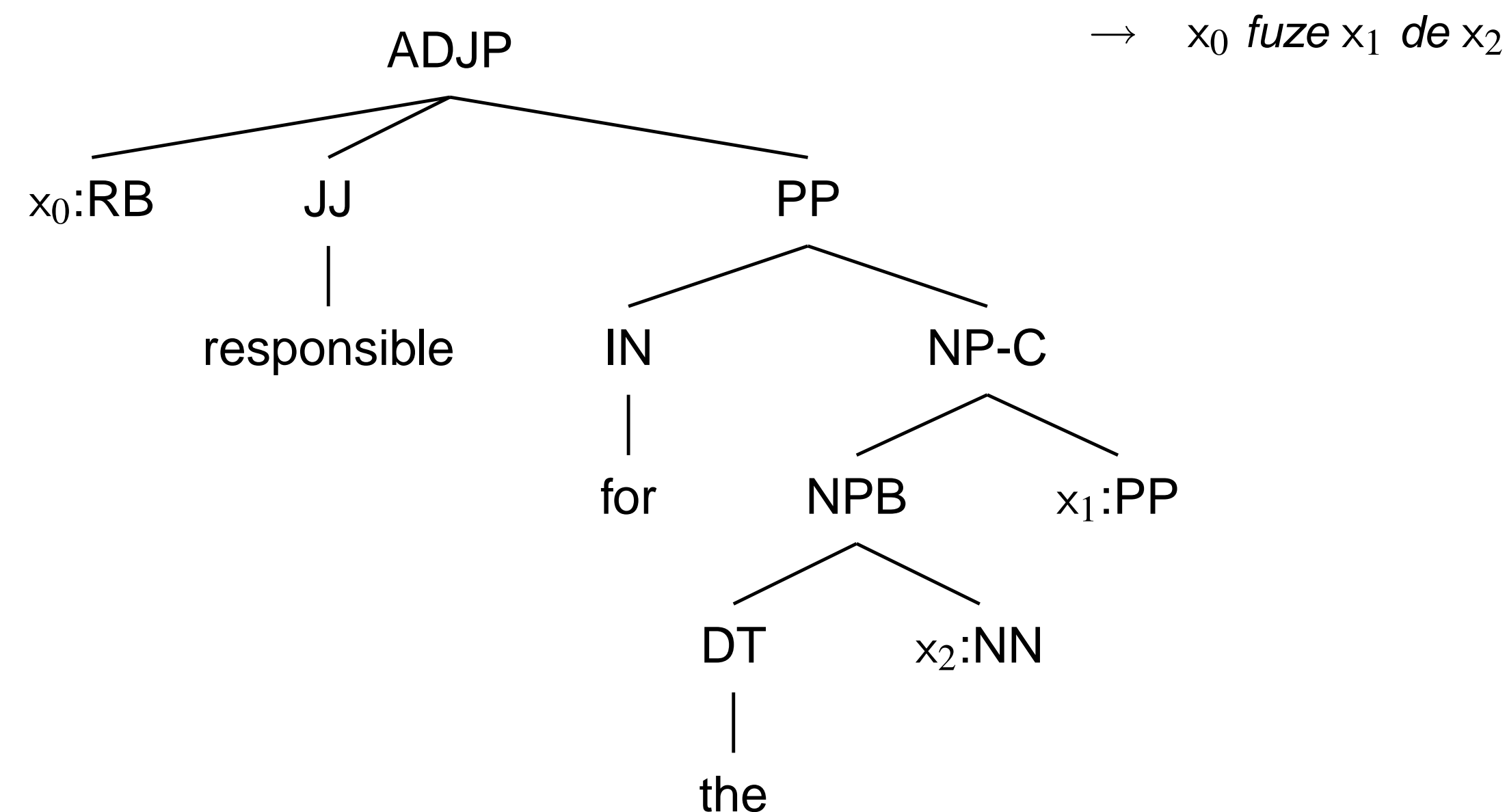


## A Shift-reduce Algorithm to Recognize Binarizable Permutations

iteration	stack	input	action
		1 5 3 4 2	
	1	5 3 4 2	shift
1	1 5	3 4 2	shift
2	1 5 3	4 2	shift
3	1 5 3 4	2	shift
	1 5 3-4	2	reduce [3,4]
	1 3-5	2	reduce <5, [3,4]>
4	1 3-5 2		shift
	1 2-5		reduce <2, <5, [3,4]>>
	1-5		reduce [1, <2, <5, [3,4]>>]



## An Actual Example: Before Binarization



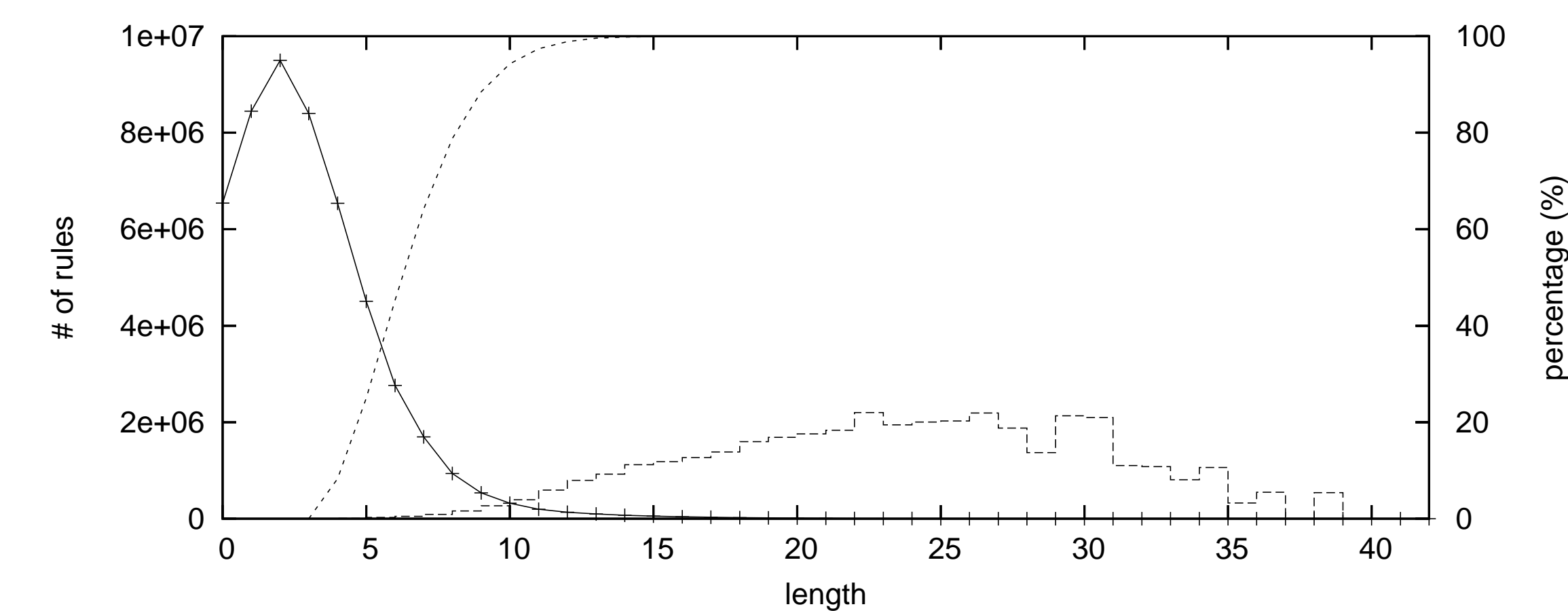
The left hand side is an English tree fragment. The right hand side is a Chinese string including coindexed variables referring to the translations of the lefthand side nonterminals.

## An Actual Example: After Binarization

ADJP  $\rightarrow T_{859}^{(1)}, T_{859}^{(1)}$   
 $T_{859} \rightarrow V_1^{(1)} V_2^{(2)}, V_1^{(1)} V_2^{(2)}$   
 $V_1 \rightarrow RB^{(1)}, RB^{(1)} fuze$   
 $V_2 \rightarrow resp. for the NN^{(1)} V_3^{(2)}, V_3^{(2)} NN^{(1)}$   
 $V_3 \rightarrow PP^{(1)}, PP^{(1)} de$

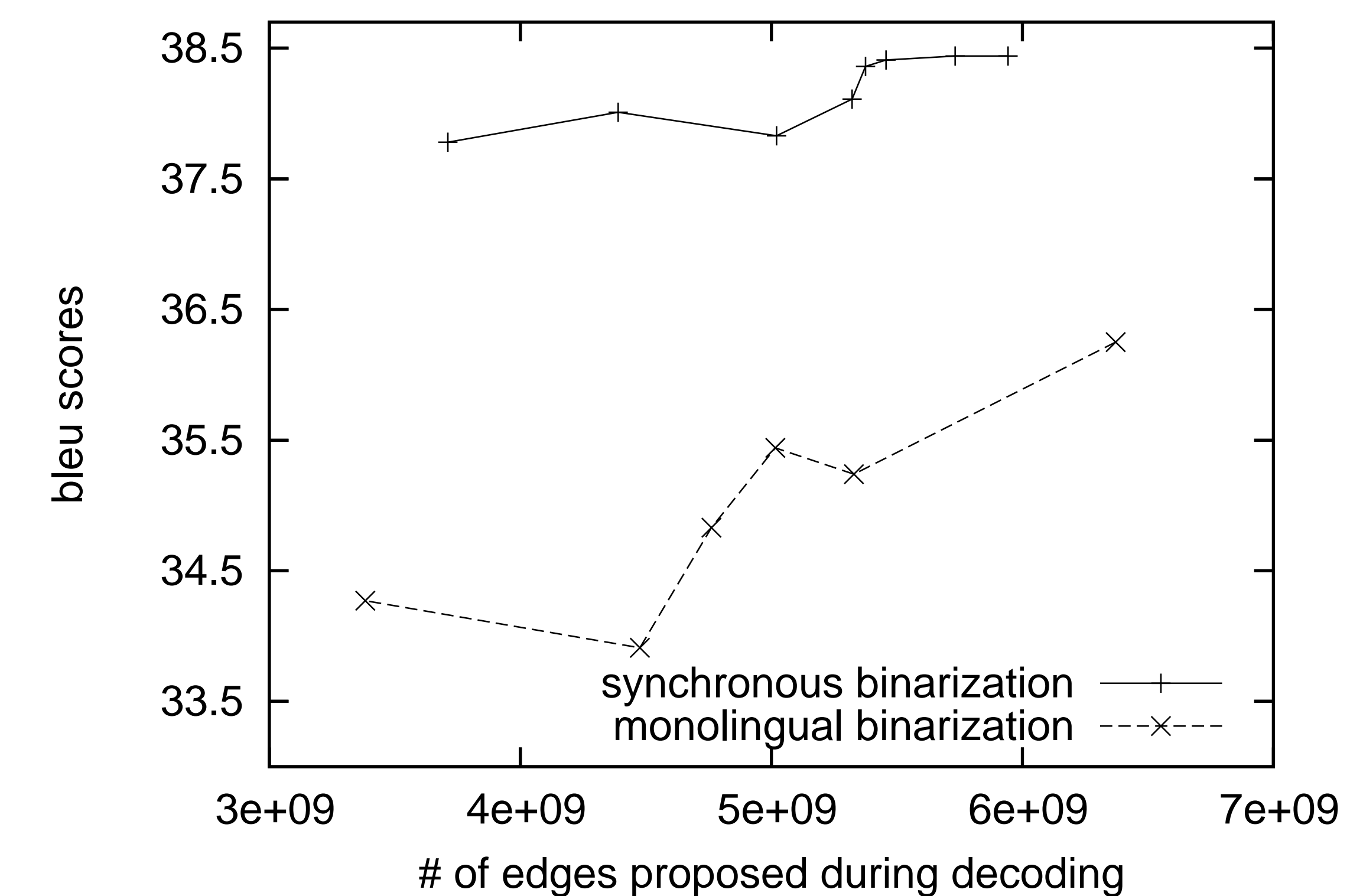
$T_{859}^{(1)}$  is a symbolic representation of the left-hand side tree in the original rule. We can reconstruct the original rule from these rules.

## How Many Rules Are Binarizable?



The solid-line curve represents the distribution of all rules against permutation lengths. The dashed-line stairs indicate the percentage of non-binarizable rules in our initial rule set while the dotted-line denotes that percentage among all permutations. We have 50,879,242 rules.

## Faster and More Accurate Decoding



## The Highest BLEU Score

system	bleu
monolingual binarization	36.25
synchronous binarization	38.44
alignment-template system	37.00

## Conclusion

- We reduced the SCFG binarization problem to the problem of hierarchical binarization of permutations and devised a linear time algorithm.
- The majority of syntactic reorderings, at least between languages like English and Chinese, can be efficiently decomposed into hierarchical binary reorderings.
- From a modeling perspective, on the other hand, it is beneficial to start with a richer representation.
- As a result, decoding with n-gram models can be fast and accurate, making it possible for our syntax-based system to overtake a comparable phrase-based system in BLEU score.