

# **Algorithms for Syntax-Aware Statistical Machine Translation**

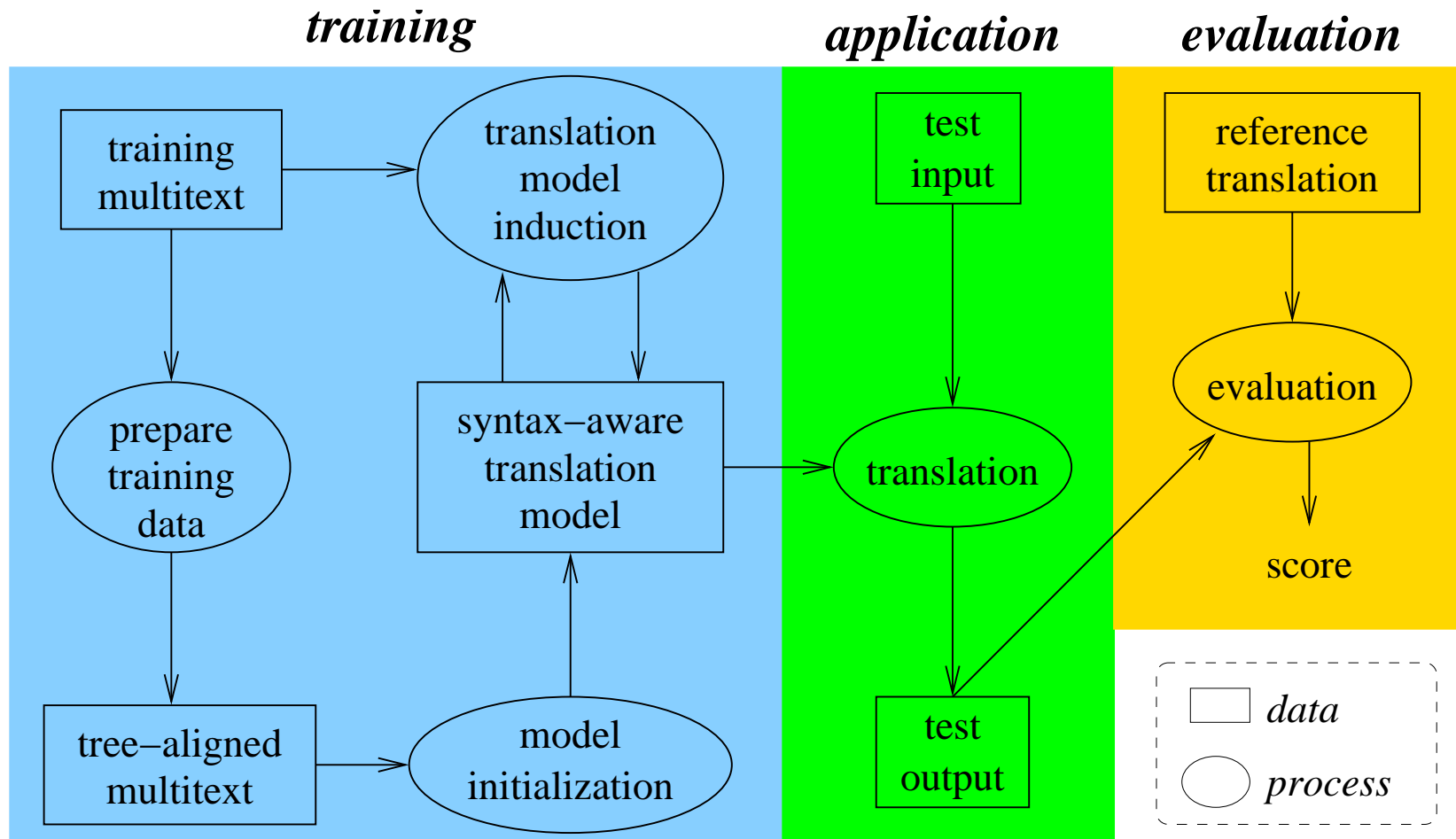
I. Dan Melamed, Wei Wang and Ben Wellington

New York University

# Syntax-Aware Statistical MT

- “Statistical”  $\Rightarrow$  involves machine learning (ML)
  - seems crucial for robustness
- “Syntax-aware”  $\Rightarrow$  involves tree structures
  - seems crucial for improving quality
- Does not preclude linguistic sophistication, but this talk will use simple examples.

# Outline

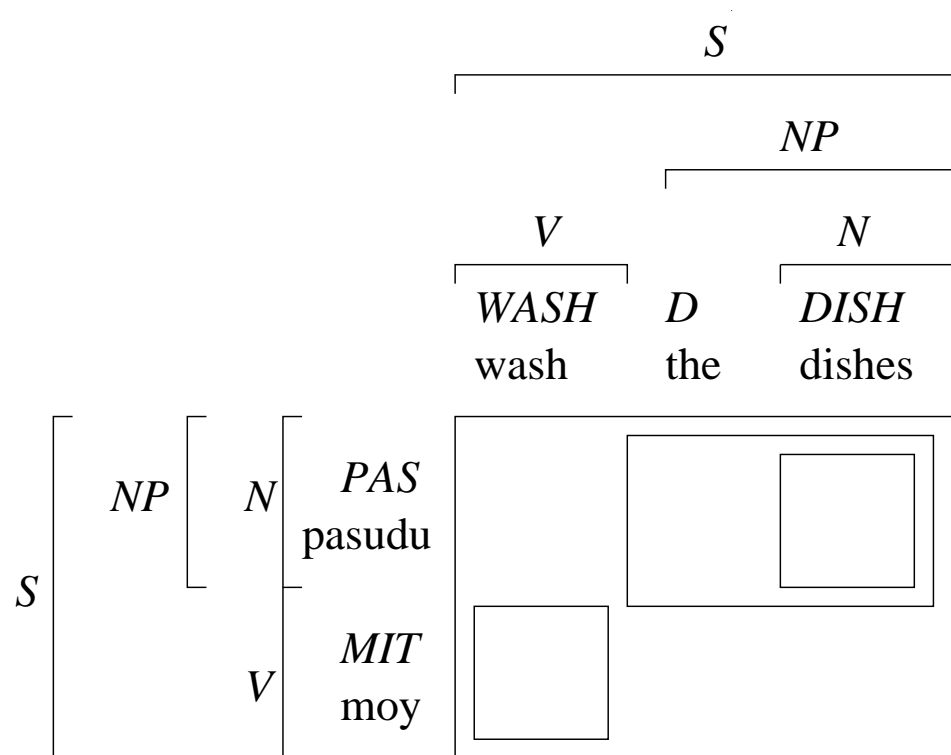


- All the non-trivial algorithms are *generalized parsers*.
- Algorithms are easier to design and implement if one understands how they are related to others.

# First Things First

- What's a generalized parser?
- What sort of grammar does it need?
- example of a generalized parser

# Generalized Parsers output Multitrees



- multi-dimensional trees
- express nested correspondence between subtrees
- ordinary trees = 1D special case

# Generalized Multitext Grammar (GMTG)

- synchronous generalization of CFG
- terminals, nonterminals, start symbol, productions
- allows discontinuous constituents (ignored here)
- admits a Generalized Chomsky Normal Form (GCNF)
- associated derivation process generates multitrees
- special cases: ITG (Wu'97), MTG (Melamed'03)

# A Simplified GMTG in GCNF

## Terminal productions

1.  $WASH \rightarrow Wash$   
 $() \rightarrow ()$
2.  $D \rightarrow the$   
 $() \rightarrow ()$
3.  $DISH \rightarrow dishes$   
 $() \rightarrow ()$
4.  $() \rightarrow ()$   
 $PAS \rightarrow Pasudu$
5.  $() \rightarrow ()$   
 $MIT \rightarrow moy$

## Nonterminal productions

1.  $S \rightarrow V NP$   
 $S \rightarrow NP V$
2.  $V \rightarrow WASH$   
 $V \rightarrow MIT$
3.  $NP \rightarrow D N$   
 $NP \rightarrow N$
4.  $N \rightarrow DISH$   
 $N \rightarrow PAS$

## Language

Wash the dishes  
Pasudu moy

# Ordinary CKY Parsing under CFG

● Input:  $0$  wash  $1$  the  $2$  dishes  $3$

● Scan: 
$$\frac{V \rightarrow \text{wash}}{V} \quad \frac{D \rightarrow \text{the}}{D} \quad \frac{N \rightarrow \text{dishes}}{N}$$
  

$$\frac{0}{0} \quad \frac{1}{1} \quad \frac{2}{2} \quad \frac{3}{3}$$

● Compose: 
$$\frac{\frac{D}{1 \quad 2} \quad \frac{N}{2 \quad 3}}{NP} \quad \frac{NP \rightarrow D \ N}{NP}$$
  

$$\frac{NP}{1 \quad 3}$$

$$\frac{\frac{V}{0 \quad 1} \quad \frac{NP}{1 \quad 3}}{S} \quad \frac{S \rightarrow V \ NP}{S}$$
  

$$\frac{S}{0 \quad 3}$$

● For probabilistic parsing, terms carry probabilities:

$$\frac{\beta \ \gamma \ \alpha \rightarrow \beta \gamma}{\alpha} \text{ means}$$

$$\Pr(\alpha) = \Pr(\beta) \cdot \Pr(\gamma) \cdot \Pr_G(\alpha \rightarrow \beta \gamma)$$

# CKY Multiparsing under GMTG

- Here only in 2D, for simplicity.

- Input:  $0$  wash  $1$  the  $2$  dishes  $3$   
 $0$  pasudu  $1$  moy  $2$

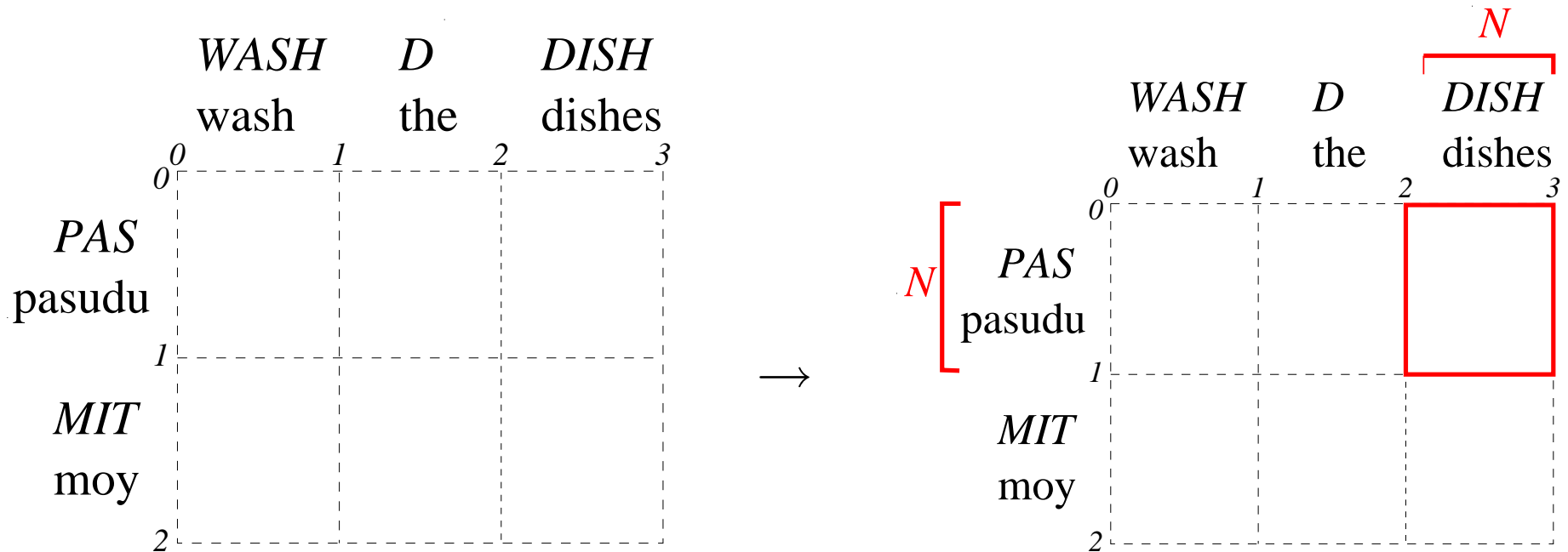
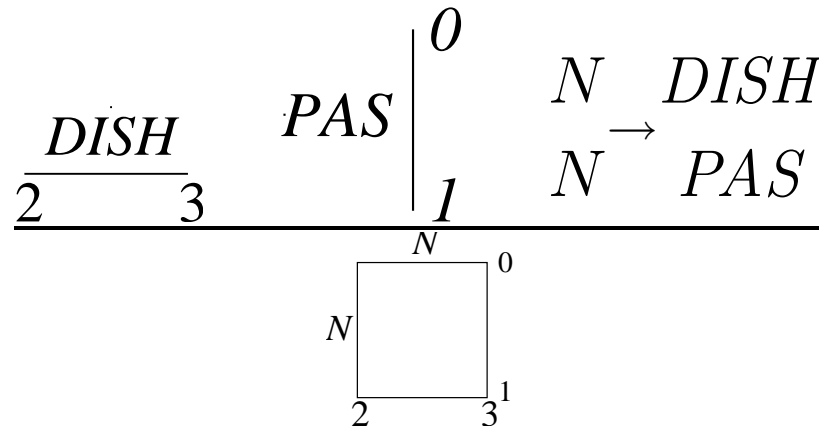
- Scan Component 1:

$$\begin{array}{ccc}
 \begin{array}{c}
 WASH \quad \text{wash} \\
 \rightarrow \\
 () \quad () \\
 \hline
 WASH \\
 \hline
 0 \quad 1
 \end{array} &
 \begin{array}{c}
 D \quad \text{the} \\
 \rightarrow \\
 () \quad () \\
 \hline
 D \\
 \hline
 1 \quad 2
 \end{array} &
 \begin{array}{c}
 DISH \quad \text{dishes} \\
 \rightarrow \\
 () \quad () \\
 \hline
 DISH \\
 \hline
 2 \quad 3
 \end{array}
 \end{array}$$

- Scan Component 2:

$$\begin{array}{cc}
 \begin{array}{c}
 () \quad () \\
 \rightarrow \\
 PAS \quad \text{pasudu} \\
 \hline
 PAS \left| \begin{array}{l} 0 \\ 1 \end{array} \right.
 \end{array} &
 \begin{array}{c}
 () \quad () \\
 \rightarrow \\
 MIT \quad \text{moy} \\
 \hline
 MIT \left| \begin{array}{l} 1 \\ 2 \end{array} \right.
 \end{array}
 \end{array}$$

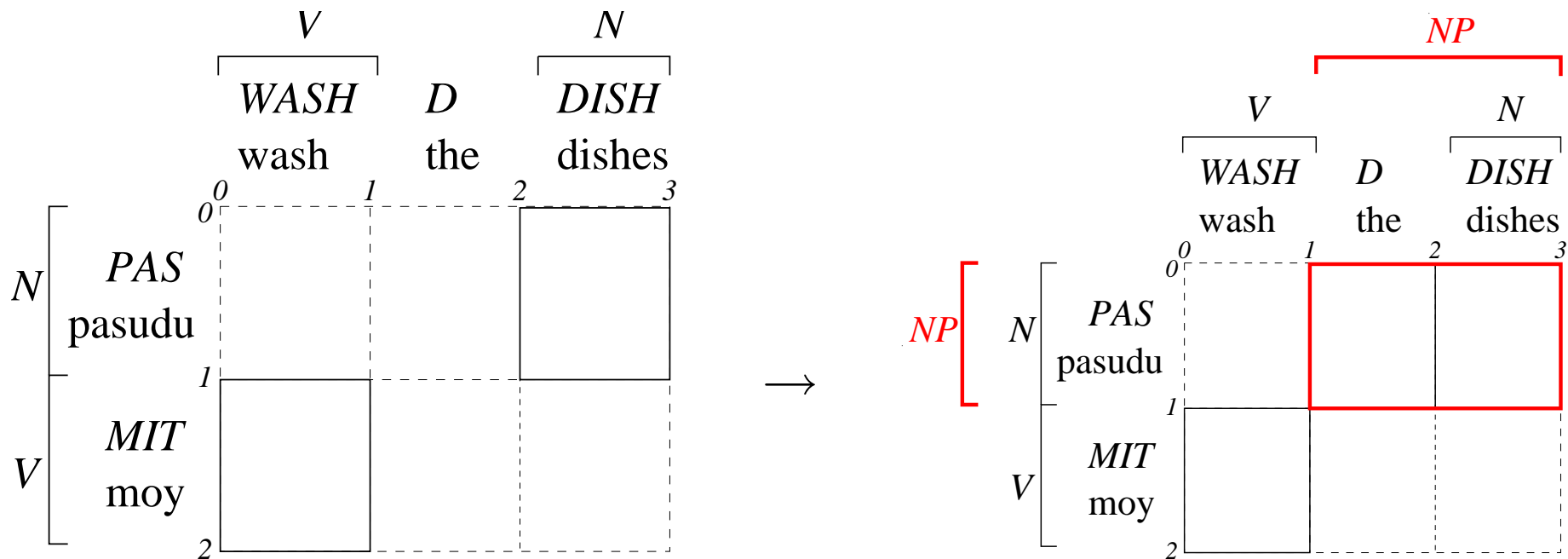
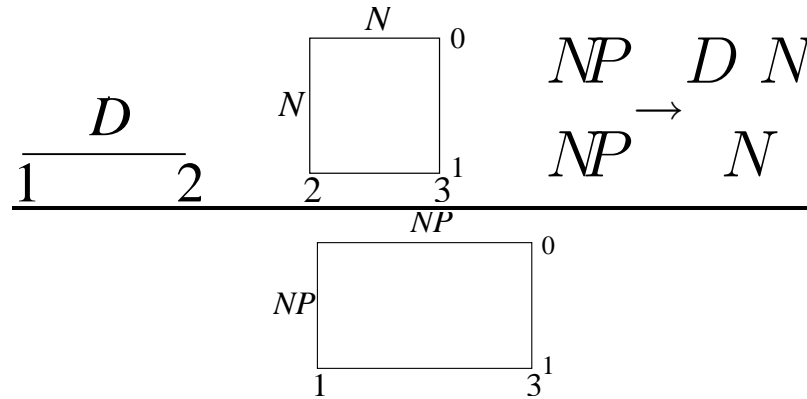
# Multiparsing: Composition (1/4)



Two 1D items can be composed into a 2D item.

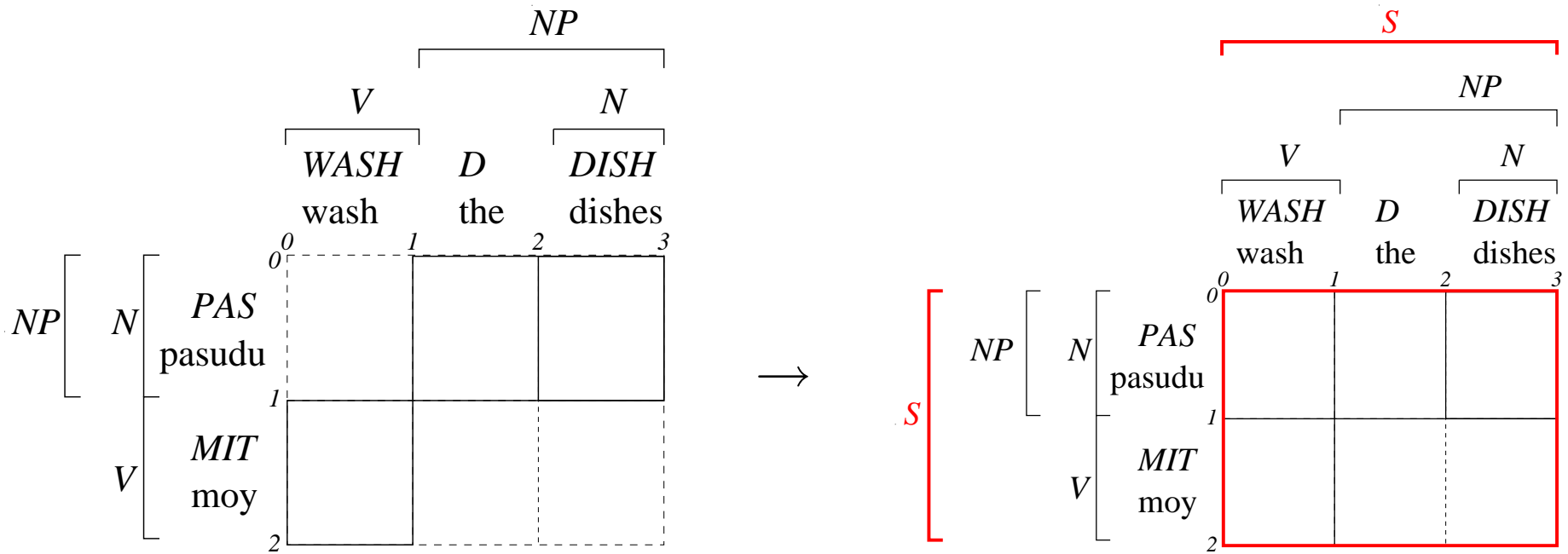
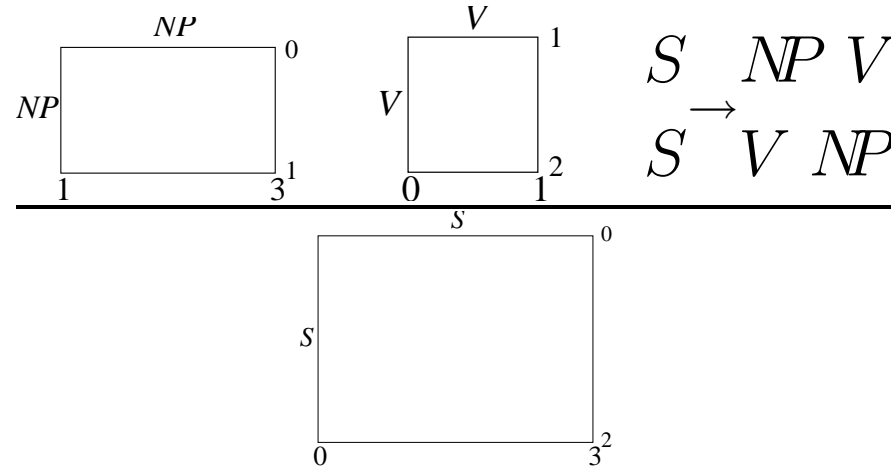


# Multiparsing: Composition (3/4)

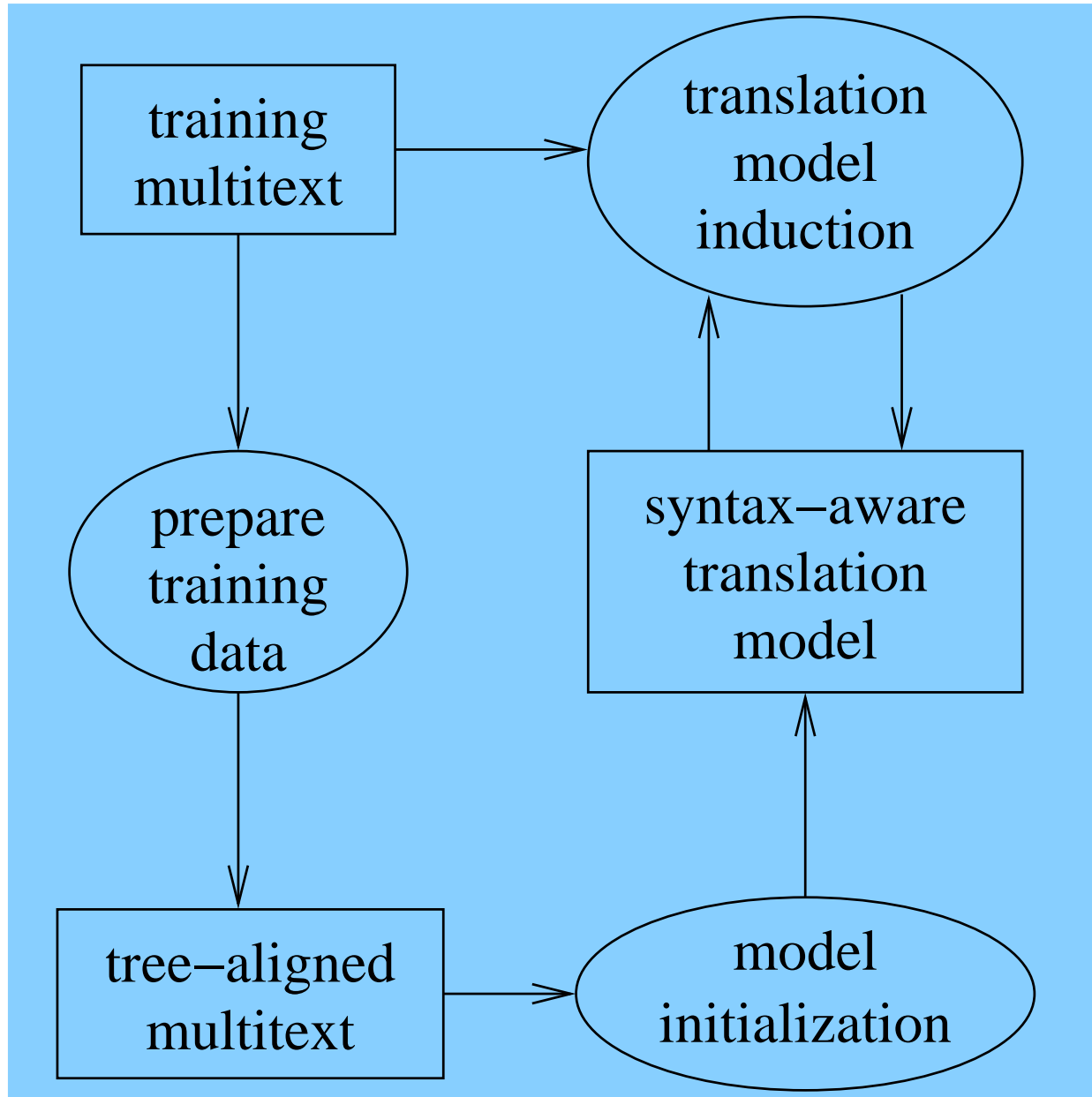


2D items can compose with 1D items.

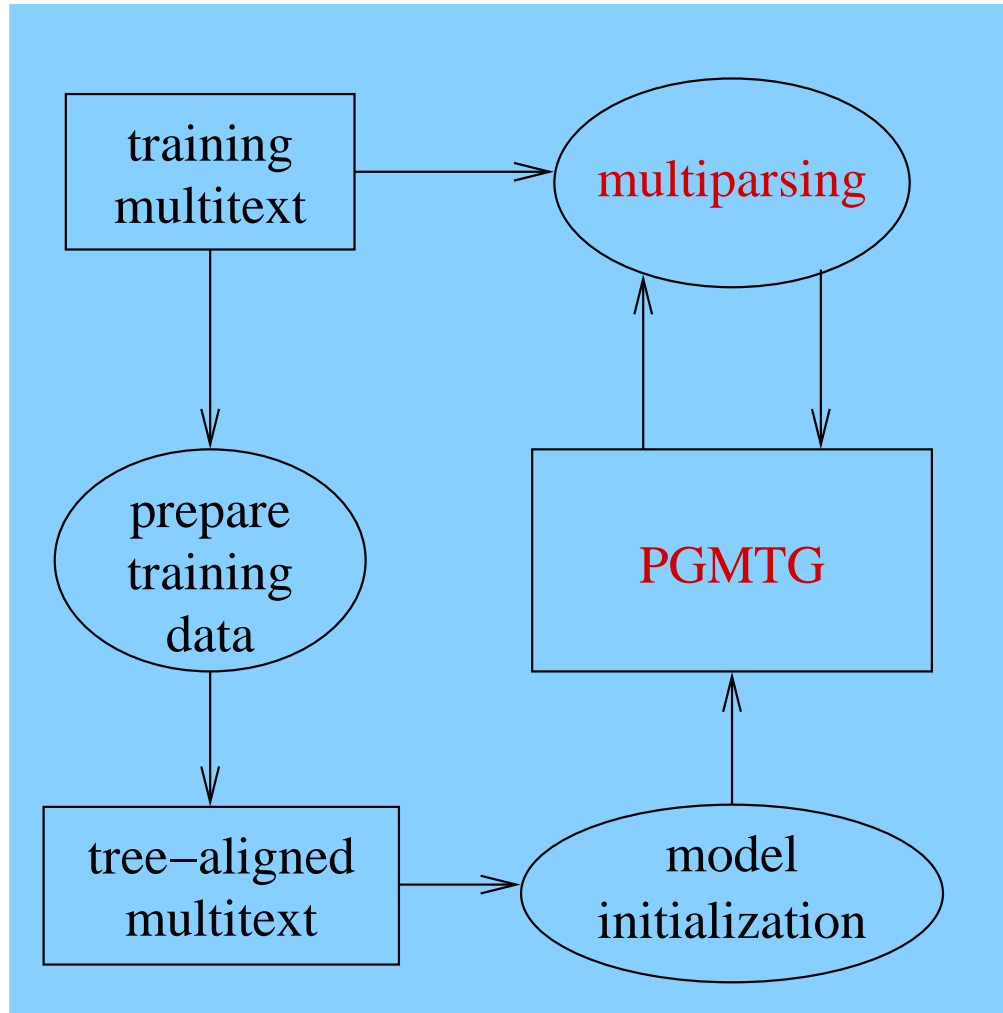
# Multiparsing: Composition (4/4)



# Part 1: Training



# TM Re-estimation by Parsing



PGMTG is a type of syntax-aware translation model.

# PGMTG Parameter Estimation

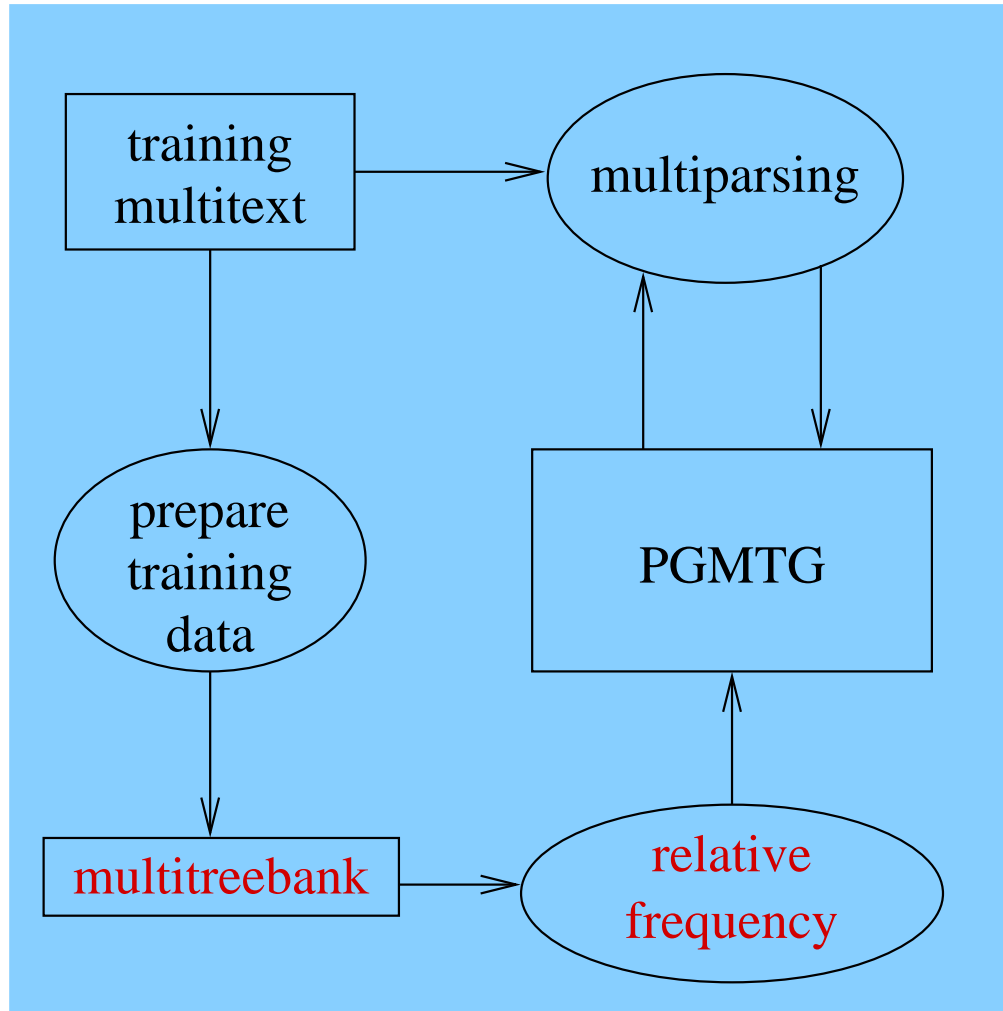
- use expectation semiring (Eisner'02)

$$\frac{\beta \quad \gamma \quad \alpha \rightarrow \beta\gamma}{\alpha} \text{ means}$$

$$E(\alpha) = E(\beta) \otimes E(\gamma) \otimes G(\alpha \rightarrow \beta\gamma)$$

- to estimate production rule probabilities:
  1. use same inference algorithm with exp-counts instead of probabilities
  2. normalize
- multiparsing + trivial normalization  
= a synchronous generalization of the Inside-Outside algorithm (cf. Wu'97 for SITG)
- multiparsing also required for other loss functions  
— e.g., max-margin à la Taskar et al.'04

# TM Induction by Parsing



How to bootstrap a multitreebank?

# Bootstrapping

- need multitreebank to initialize PGMTG
- multitreebank  $\neq$  parallel treebank (Han et al.'02)
- no multitreebanks currently available  
(though Uchimoto et al.'04 working on one)
- need generalized parser to create multitreebank
- parser needs to know values that  $G$  assigns to productions in order to compute  $\frac{\beta \ \gamma \ \alpha \rightarrow \beta \gamma}{\alpha}$
- Stuck?

# How to bootstrap $G()$ values?

- Consider the bilexical case. Ordinarily, we have

$$G \left( \begin{array}{c} S[\text{wash}] \\ S[\text{moy}] \end{array} \rightarrow \begin{array}{c} V[\text{wash}] \text{ NP}[\text{dishes}] \\ V[\text{moy}] \text{ NP}[\text{pasudu}] \end{array} \right) =$$
$$\Pr \left( \begin{array}{c} V[\text{wash}] \text{ NP}[\text{dishes}] \\ V[\text{moy}] \text{ NP}[\text{pasudu}] \end{array} \middle| \begin{array}{c} S[\text{wash}] \\ S[\text{moy}] \end{array} \right)$$

- To bootstrap,
  - Decompose via chain rule.
  - Make independence assumptions, so that

$$\Pr \left( \begin{array}{c} B[a] \ C[c] \\ Y[x] \ Z[z] \end{array} \middle| \begin{array}{c} A[a] \\ X[x] \end{array} \right) = \Pr(B, C, c|A, a) \times 1.0 \times \Pr(z|c)$$

# How to bootstrap $G()$ values?

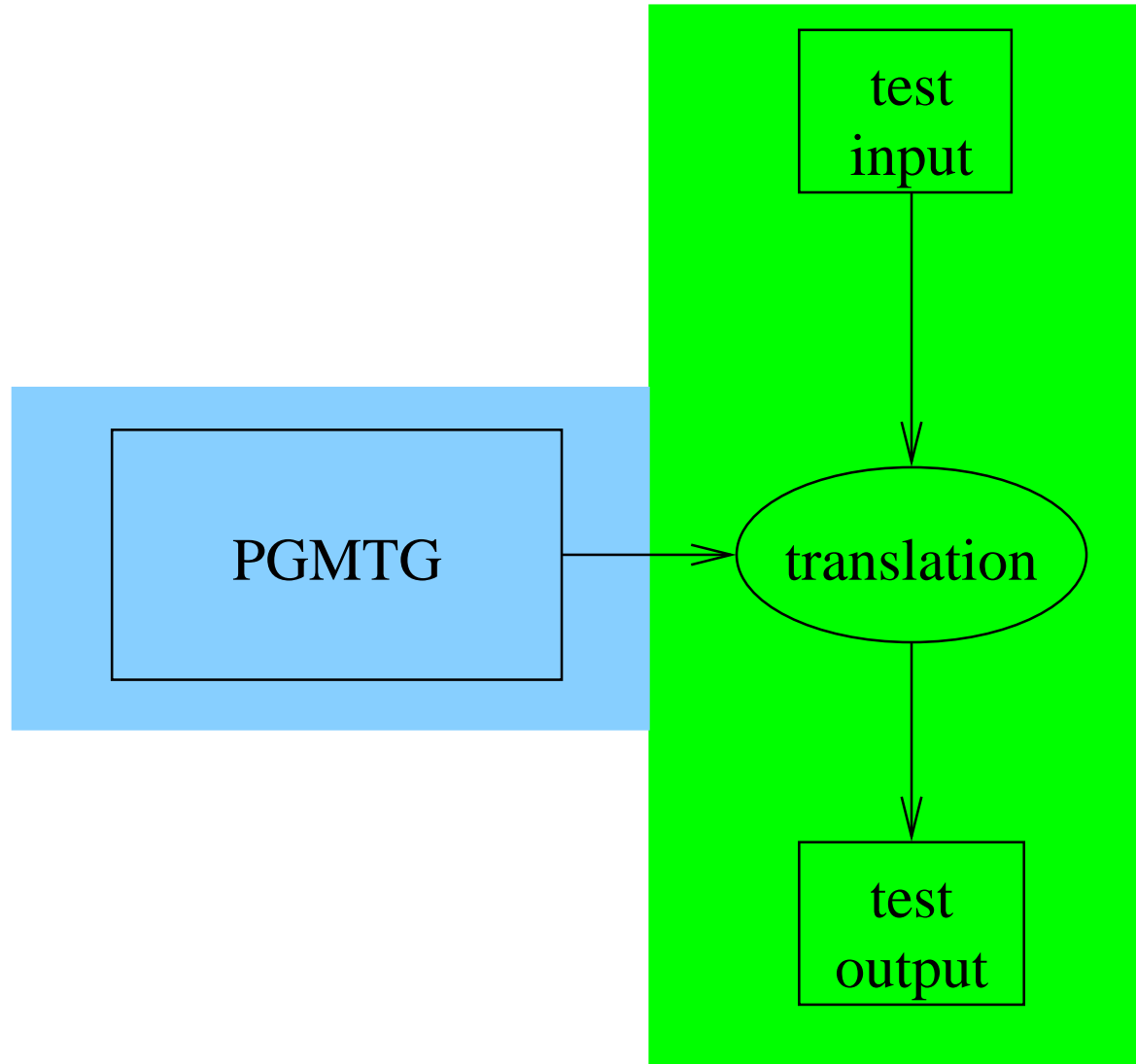
- $G \left( \begin{array}{c} A[a] \\ X[x] \end{array} \rightarrow \begin{array}{cc} B[a] & C[c] \\ Y[x] & Z[z] \end{array} \right) = \Pr(B, C, c|A, a) \times \Pr(z|c)$
- $\Pr(B, C, c|A, a)$  is an ordinary (monolingual) structured language model, such as lexicalized PCFG.
- $\Pr(z|c)$  is a word-to-word translation model.
- These resources are relatively easy to obtain.
- Now, we can use the same inference procedure!
- $$\frac{\beta \ \gamma \ \alpha \rightarrow \beta\gamma}{\alpha}$$
- a multitext aligner is a generalized parser, where  $\dim(\text{grammar}) \leq \dim(\text{input})$



# Other aligners

- More sophisticated aligners certainly possible  
— e.g. Smith & Smith @ EMNLP'04

# Part 2: Application of the Model



# Translation: Scanning

● Input:     <sub>0</sub> wash <sub>1</sub> the <sub>2</sub> dishes <sub>3</sub>

● Scan Component 1, as usual:

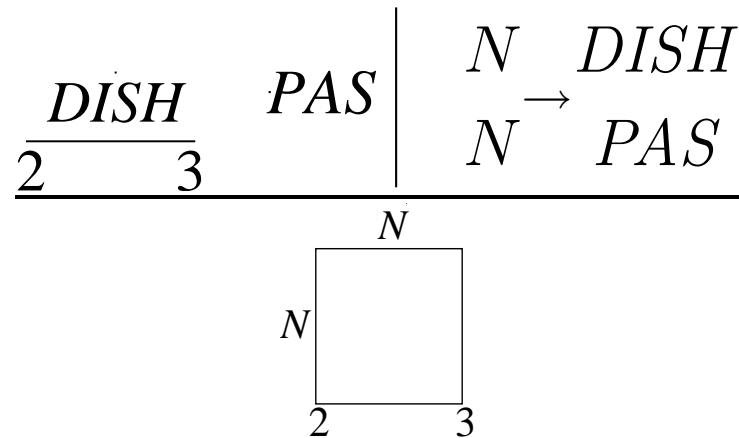
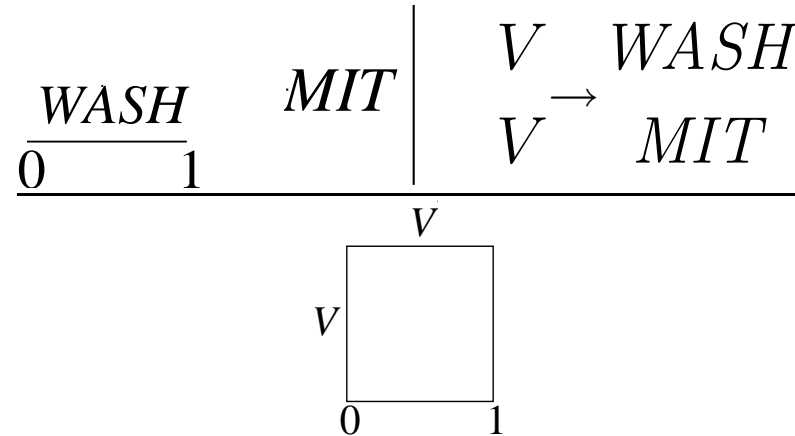
$$\begin{array}{ccc} \begin{array}{c} WASH \rightarrow \text{wash} \\ () \quad \rightarrow \quad () \\ \hline WASH \\ \hline \underline{0} \quad \underline{1} \end{array} & \begin{array}{c} D \rightarrow \text{the} \\ () \quad \rightarrow \quad () \\ \hline D \\ \hline \underline{1} \quad \underline{2} \end{array} & \begin{array}{c} DISH \rightarrow \text{dishes} \\ () \quad \rightarrow \quad () \\ \hline DISH \\ \hline \underline{2} \quad \underline{3} \end{array} \end{array}$$

● Scan (Load) Component 2, independent of the input:

$$\begin{array}{cc} \begin{array}{c} () \rightarrow () \\ MIT \rightarrow \text{moy} \\ \hline MIT \\ \hline \end{array} & \begin{array}{c} () \rightarrow () \\ PAS \rightarrow \text{pasudu} \\ \hline PAS \\ \hline \end{array} \end{array}$$

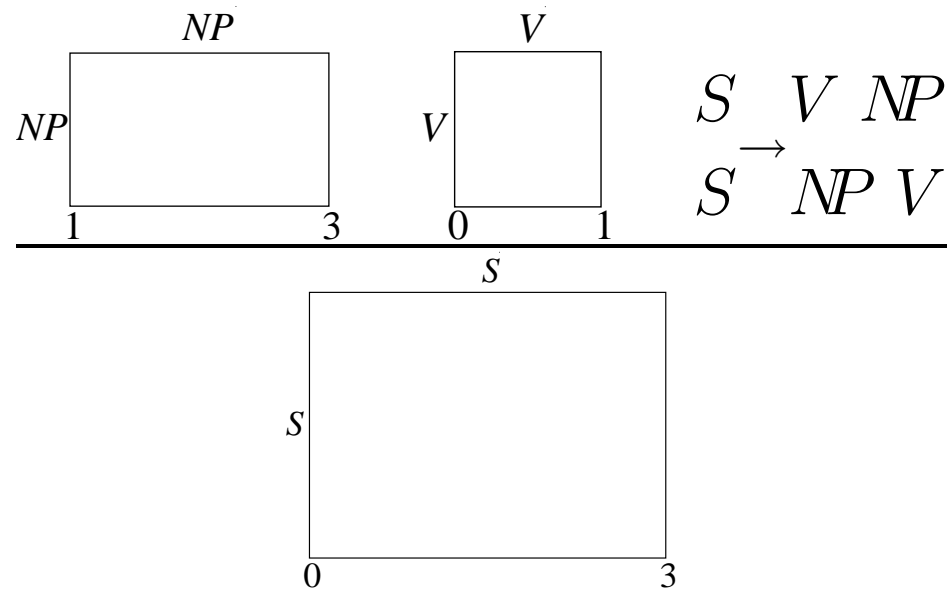
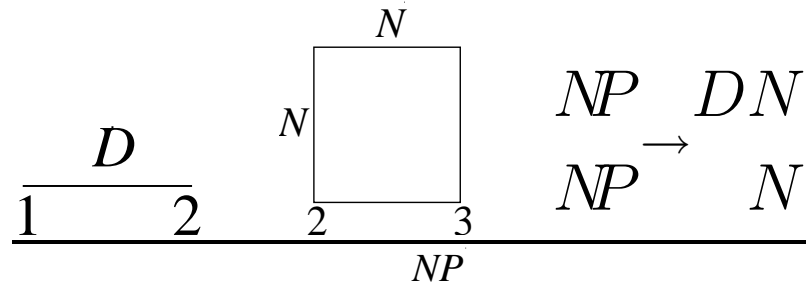
● No position info stored in 2nd component.

# Translation: Composition (1/2)

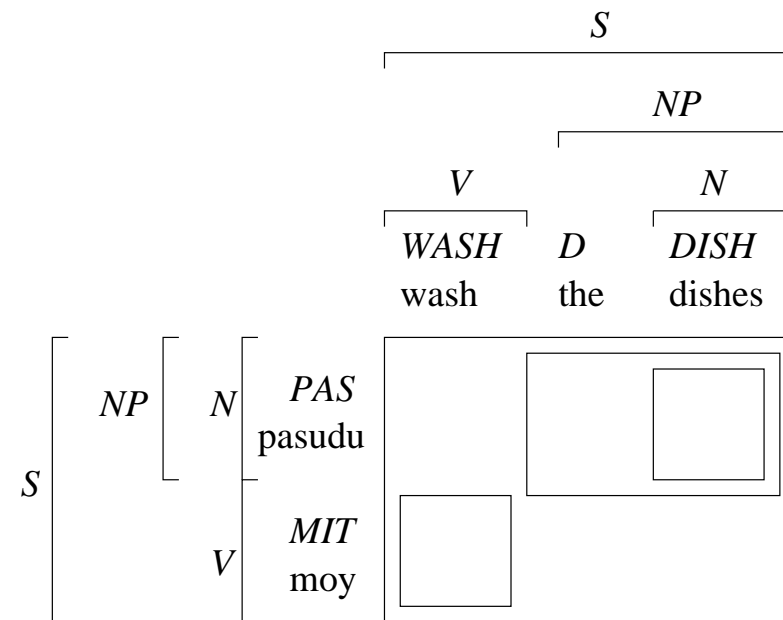


The grammar also functions as a translation model.

# Translation: Composition (2/2)



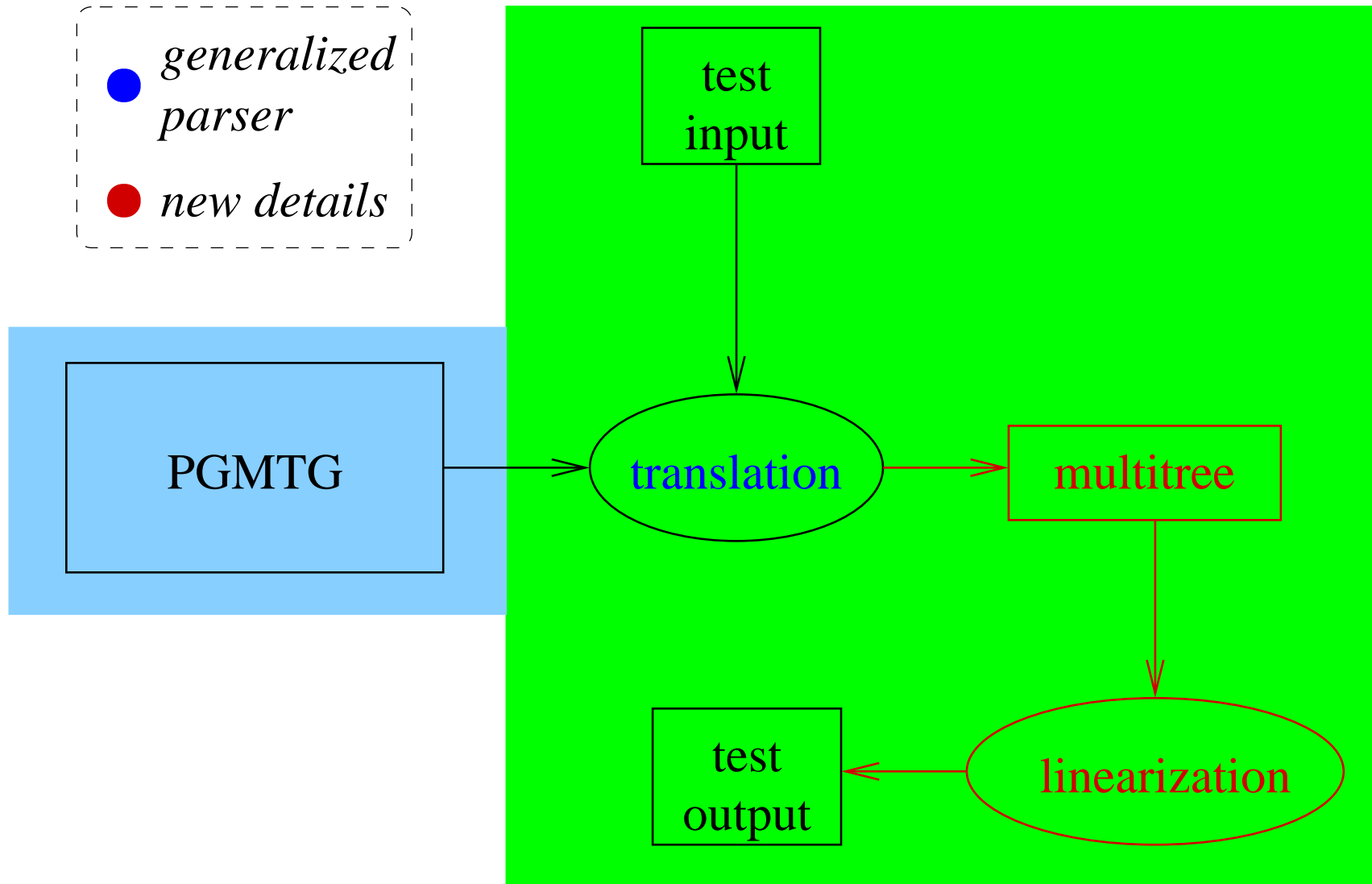
# Translation: Linearization



To recover the output string from the multitree:

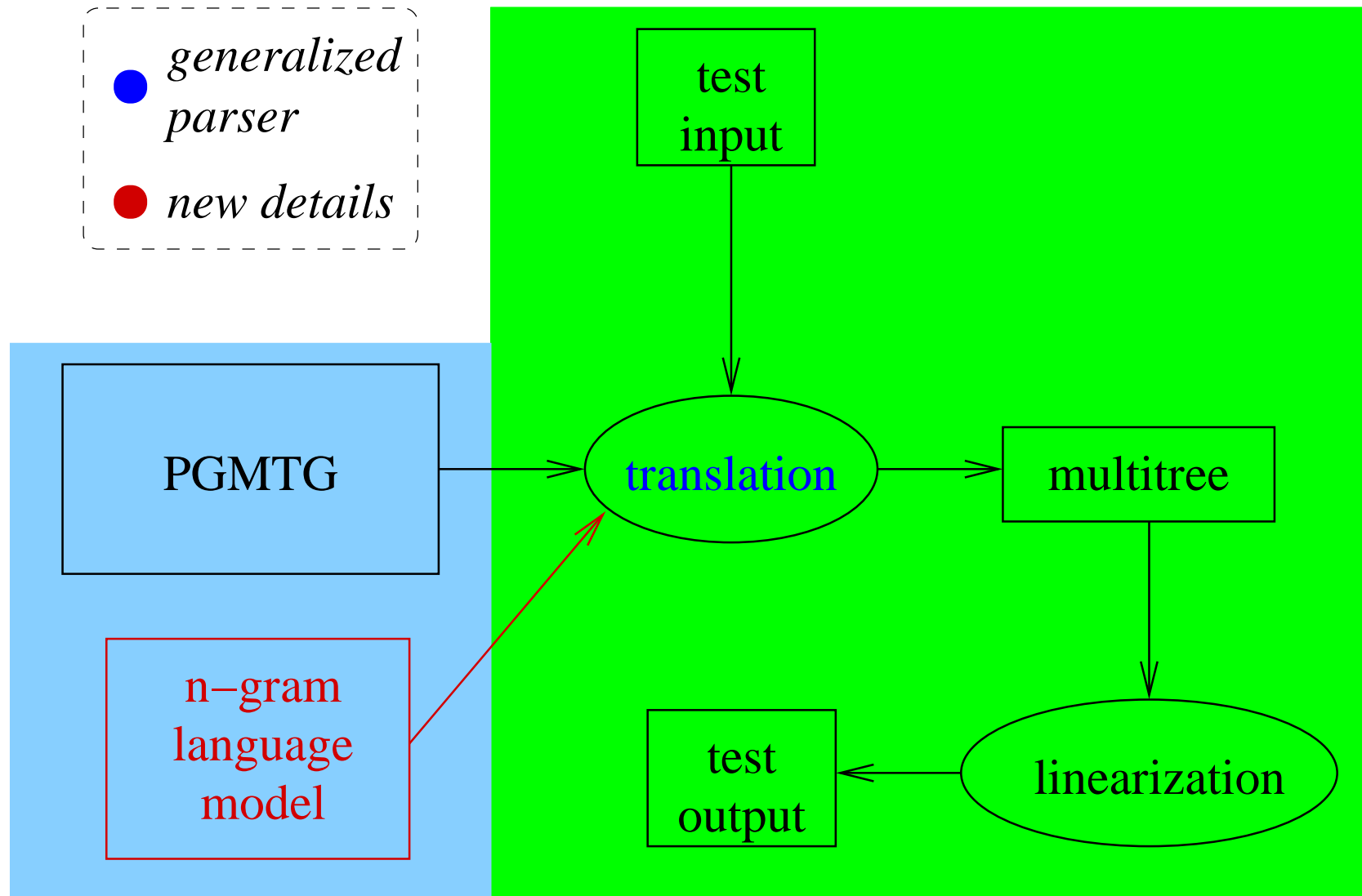
- keep track of inferences used  
(automatic under Viterbi-derivation semiring)
- recursively order the output words according to the production rules in those inferences.
- — a trivial linear-time post-process

# Translation by Parsing



A translator is a generalized parser where  
 $\dim(\text{grammar}) \geq \dim(\text{input})$ .

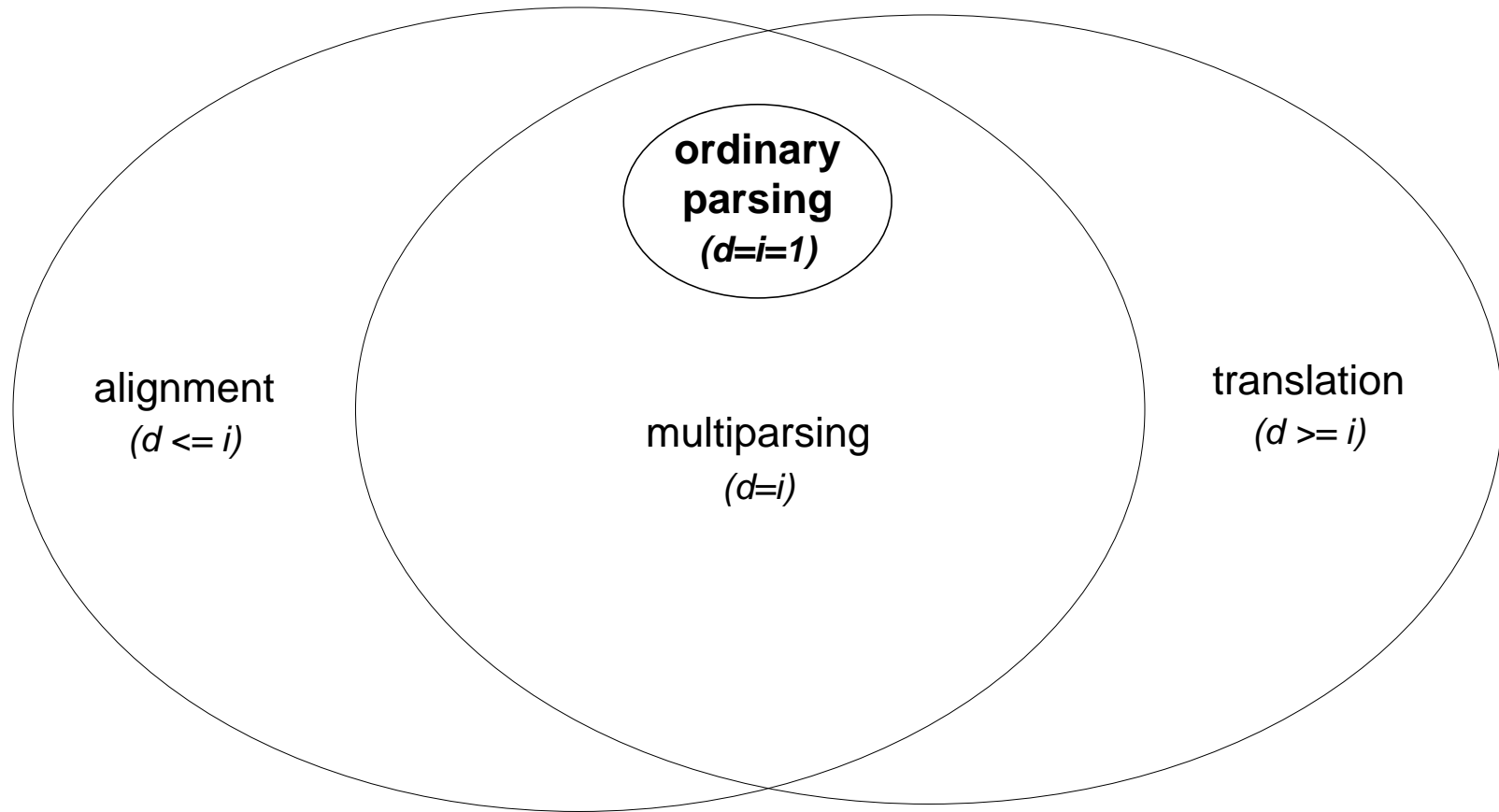
# Translation by Parsing



To incorporate a target LM, need only to redefine  $G()$  terms.

# The Space of Generalized Parsers

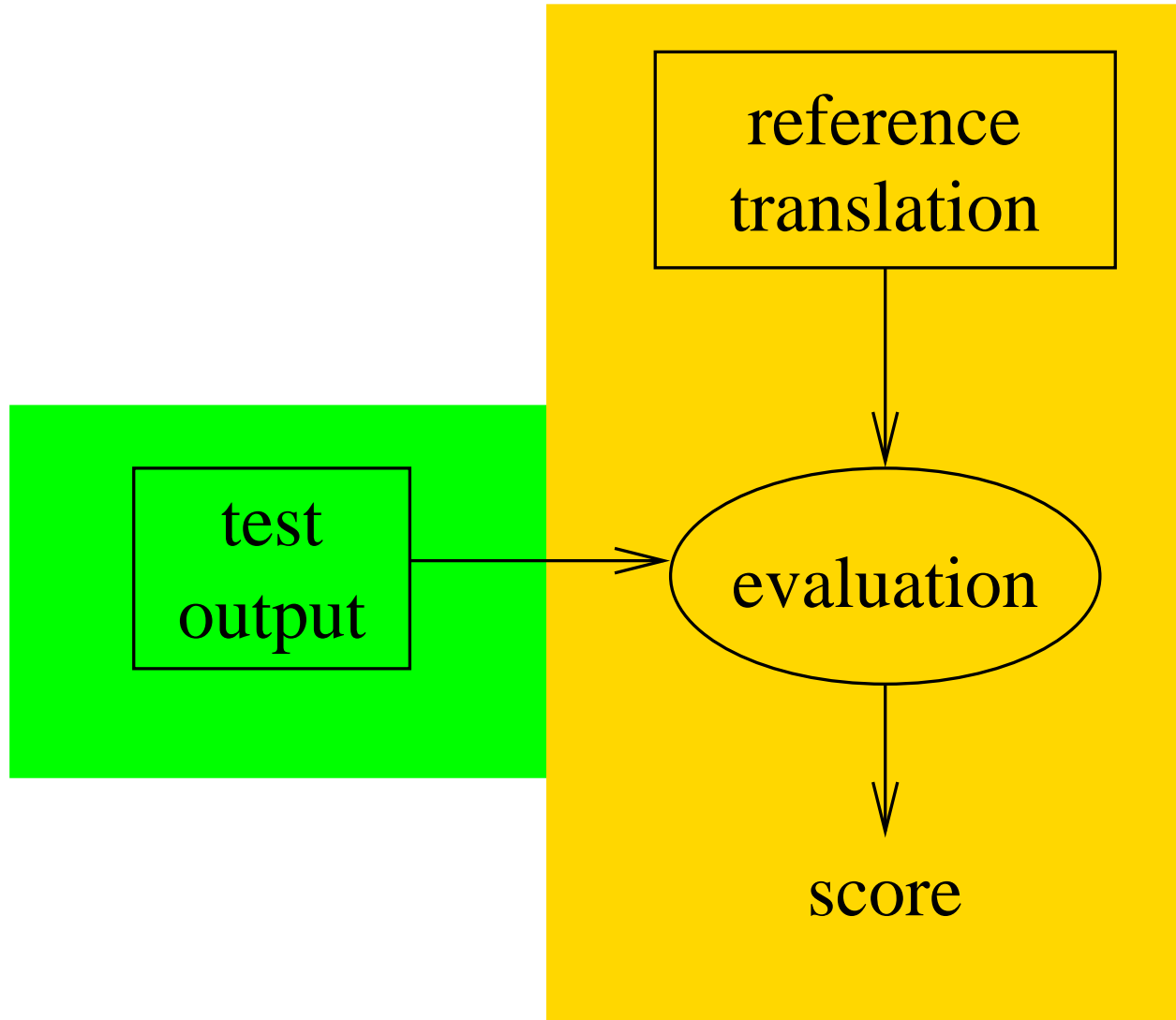
Generalized parsers infer multitrees.



$$d = \dim(\text{grammar}); i = \dim(\text{input})$$

Possible to use the same piece of code for all  $d$  and  $i$ !

# Part 3: Automatic Evaluation



# Problems with Existing Methods

- (R) *Pat asked Sandy on Friday about the man from Oslo.*
- (T1) *On Friday, Pat asked Sandy about the man from Oslo.*
- (T2) *Pat from Oslo asked Sandy on Friday about the man.*
- String-based evaluation methods prefer T2, because it has a longer-matching n-gram with R.
  - Methods that don't refer to R cannot tell the difference – T1 and T2 are both grammatical.
  - More sophisticated MT systems require more sophisticated evaluation methods.

# Syntax-Aware MT Evaluation

- catalogue of meaning-preserving syntactic alternations
- easy to express using synchronous production rules
- e.g., include the production

$$\begin{array}{l} \text{NP} \quad \text{NN PP}_1 \text{ PP}_2 \\ \text{NP} \xrightarrow{\quad} \text{NN PP}_2 \text{ PP}_1 \end{array}$$

- also include all productions that have identical components in both dimensions, such as

$$\begin{array}{l} \text{NP} \quad \text{NN PP}_1 \text{ PP}_2 \\ \text{NP} \xrightarrow{\quad} \text{NN PP}_1 \text{ PP}_2 \end{array}$$

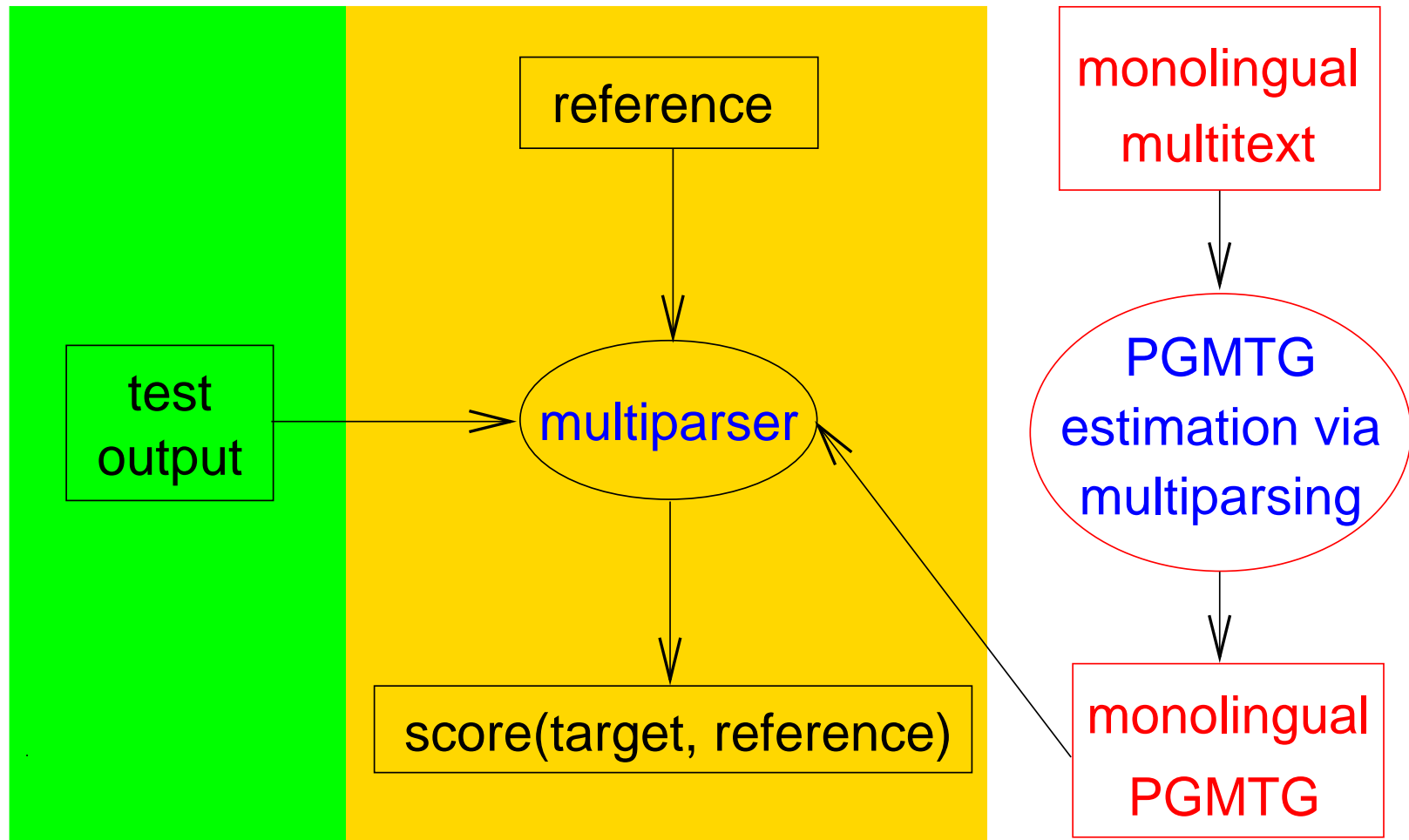
# MT Evaluation by Parsing – Ideal World

- Given
  - reference R
  - MT output T
  - grammar of syntactic alternations G
- T is an acceptable translation w.r.t. R and G iff the multitext (R, T) is parsable under G.
- Parser will fail if
  - T and R mean different things, or
  - T is ungrammatical.

# MT Evaluation by Parsing – Real World

- Challenges:
  - Hard to get good coverage of syntactic alternations by hand.
  - Would like a numerical grade of translation quality, not just a yes/no.
- Solution 0 (Leutsch et al., 2003): Use trivial grammar with 1 non-terminal.
- Solution 1: Use multiple-translations corpora to estimate a monolingual PGMTG.
- A *probabilistic* multiparser can then return the *probability* that T and R mean the same thing.
- Imperfect, but probably better than current methods.
- Competing multitext models can be evaluated in terms of their perplexity on held-out multitext.

# MT Evaluation by Parsing



● *generalized parser*      ● *new details*

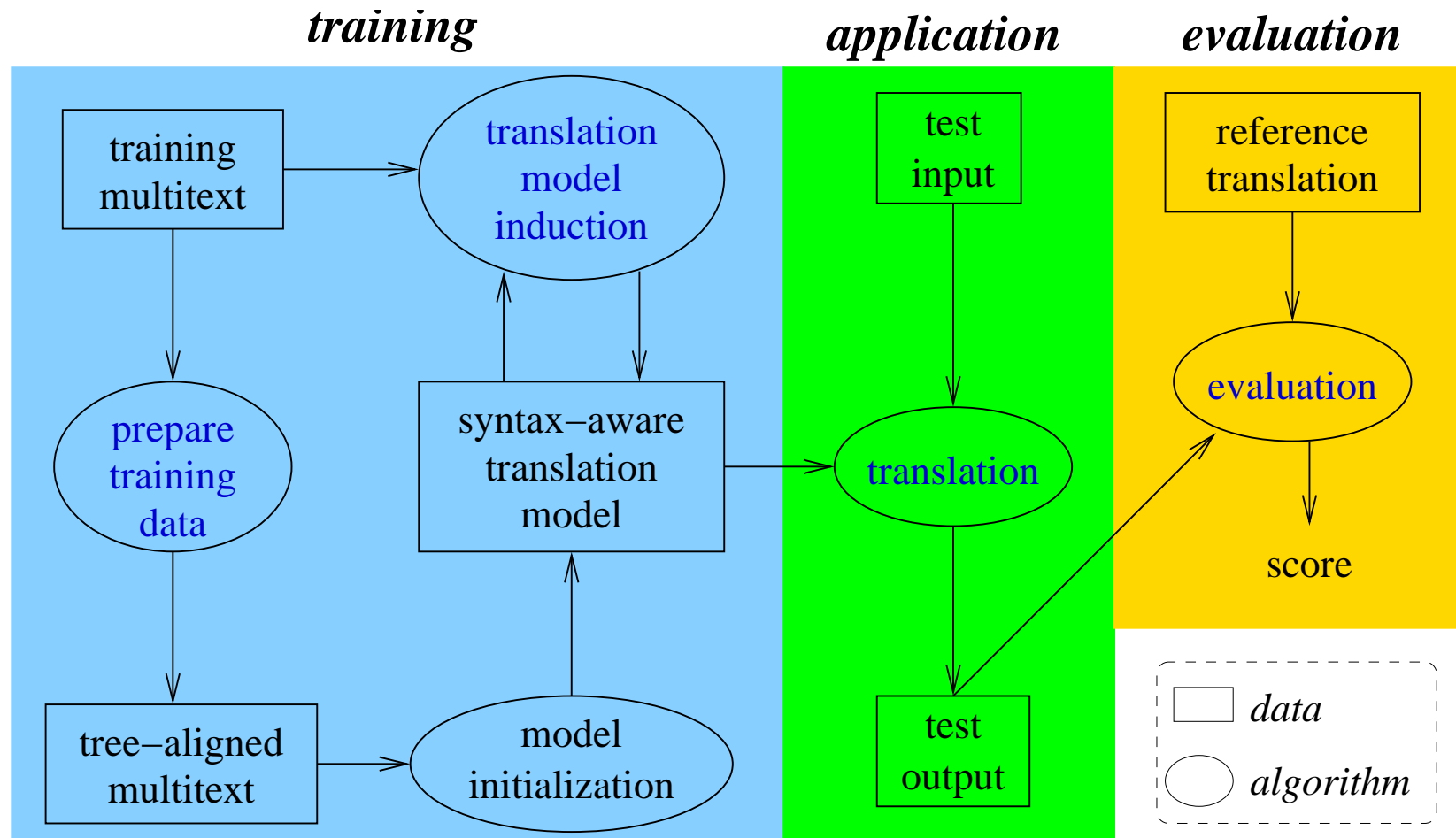
All the non-trivial algorithms are *generalized parsers*.

# MT Evaluation by Parsing

Advantages over existing methods:

- Sensitive to meaning-preserving syntactic alternations.
- Multiple references conceptually straightforward.
- The evaluation method itself can be objectively evaluated, without relying on human judgments.

# Summary



- All the non-trivial algorithms are *generalized parsers*.
- Algorithms are easier to design and implement if one understands how they are related to others.

# Accurate SMT by Parsing

will need better

- knowledge representations
  - GMTG is not the final word
  - e.g., synch. tree substitution grammars (Eisner'03)
  - have parser, will translate
- parsing logics:
  - e.g., for optimization (Melamed'03)
  - How about an Earley translator (cf. Wu & Wong'98)?
- semirings & objective functions: beyond MLE
- search strategies
  - e.g. based on  $A^*$  parsing (Klein & Manning 2003)

**Every innovation in parsing is *directly* useful for MT!**

# More Details

<http://cs.nyu.edu/~melamed/pubs.html>

Thank you!

# Multitext Modeling

- There are different ways to estimate a PGMTG.
- Which method should we use for MT evaluation?
- Quality of PGMTG  $G \propto \Pr_G(\text{unseen multitext})$ .
- Analogous to perplexity of a language model.
- **Multitext modeling** = synchronous language modeling.
- All methods of ordinary language modeling apply.
- E.g., all productions with identical components smoothed with non-zero probability.

# A common objection to SMT by Parsing

You can't really run algorithms in  $O(n^{10})$ !

- Current SMT systems are based on algorithms that run in *exponential* time, but that doesn't stop them!
- Worst-case bounds indicate the size of the search space.
- Practical algorithms do not search the whole space – they prune.
- Still, it's nice to know that PGMGTGs constrain the search better than currently popular models.

# Translation by Parsing with a Target LM

To evaluate inference rules, a naive translator evaluates

$$G \left( \begin{array}{l} S \rightarrow V[\text{wash}] \text{ NP}[\text{ dishes } ] \\ S \rightarrow V[\text{ moy } ] \text{ NP}[\text{ pasudu } ] \end{array} \right) = \Pr \left( \begin{array}{l} V[\text{wash}] \text{ NP}[\text{ dishes } ] \\ V[\text{ moy } ] \text{ NP}[\text{ pasudu } ] \end{array} \middle| \begin{array}{l} S \\ S \end{array} \right)$$

Simplifying a bit,

$$\begin{aligned} \Pr (V[\text{wash}], \text{NP}[\text{ dishes } ], V[\text{ moy } ], \text{NP}[\text{ pasudu } ] | S, S) = \\ \Pr (V[\text{ moy } ], \text{NP}[\text{ pasudu } ] | S, S) \\ \times \Pr (V[\text{wash}], \text{NP}[\text{ dishes } ] | V[\text{ moy } ], \text{NP}[\text{ pasudu } ], S, S) \end{aligned}$$

# Translation by Parsing with a Target LM

$$\begin{aligned} \Pr (V[\text{wash}], NP[\text{dishes}], V[\text{moy}], NP[\text{pasudu}] | S, S) = \\ \Pr (V[\text{moy}], NP[\text{pasudu}] | S, S) \\ \times \Pr (V[\text{wash}], NP[\text{dishes}] | V[\text{moy}], NP[\text{pasudu}], S, S) \end{aligned}$$

- 2nd factor: conditionalized PGMTG
- 1st factor: *n-gram grammar*
- encapsulates ordinary n-gram language model
- adds facility to compose discontinuous strings
- mixture need not be uniform

# SMT by Parsing vs. Tree Transducers

- Tree transduction is the special case of synchronous parsing where the inferences are constrained by a tree on one component.
- Past and future work on parsing (more) easily generalizable, using the approach described here.
- See Klein & Manning (EMNLP'02) on conditional structure vs. conditional estimation.