Statistical Machine Translation

Searching in statistical machine translation

Approaches and efficiency/quality

STATISTICAL MACHINE TRANSLATION

Functionality

Find most probable sentence in target language for sentence in source language

Automatically align words and phrases within sentence pairs in parallel corpus

Probabilities are determined automatically by training against parallel corpus

Advantages

Can deal with lexical ambiguities and idioms (in some way)

Requires minimal human effort

Can be created for any language pair that has enough training data

Disadvantages

Does not explicitly deal with syntax (in basic models)

THE BASIC MODEL

Probablities

Find most probable sentence in target language for sentence in source language

p(*t* | *s*) = (p(*t*) * p(*s* | *t*)) / p(*s*) (maximization of p(*t*) independent of p(*s*))

p(*t*) – *the "Language model"*

Higher probabilities for fluent / grammatical sentences

Estimated by monolingual corpora

Standard is a trigram language model, calculated with probabilistic grammar

p(*s* | *t*) – *the "Translation model"*

Higher probabilities for sentences that have corresponding meaning

Estimated by bilingual corpora

THE TRANSLATION MODEL

Functionality

Define word correspondences across languages (for 1 pair of languages)

Including 1:1, N:1, 1:N alignments, additions and deletions

Submodels (e.g., IBM Model 4)

Fertility of a word – number of words it is aligned to

For reordering, distinguishing in target language between

- **Heads (leftmost word generated from some source language word)**
- **Non-heads (generated by very fertile source language words)**
- **NULL-generated(extra word without counterpart in source language)**

Probability of a sentence is the product of the probabilities of the submodels

DECODING – A SEARCH PROBLEM

Functionality

Takes previously unseen sentence in source language *s*

Searches a sentence in the target language *t* maximizing p(*t* | *s*) (p(*t*) * p(*s* | *t*))

Complexity – depending on the flexicility of reordering operations

No reordering – linear Viterbi algorithm suffices

Local reordering (around nodes in a binary tree) – high-polynomial algorithms

Arbitrary reordering – NP-complete

Some decoding strategies

Stack-based decoding

Greedy decoding

Integer programming decoding

DECODING – SOME TECHNIQUES (1)

Motivation

Search space explodes exponentially in dependency of sentence length

Beam search applied

Hypothesis recombination

Different paths lead to the same partial translation

Combine paths by dropping the weaker hypothesis, but keep pointer

Also possible with incomplete match, if:

- **Last two English words match (language model, trigrams)**
- **Foreign word coverage vectors match (effects future path)**

DECODING – SOME TECHNIQUES (2)

Motivation

Hypothesis recombination insufficient

Pruning

Heuristically discard weak hypotheses

Organize hypotheses in stack

- **Same foreign words covered or same number of target words**

Compare hypotheses in stack, discard bad ones

- **Histogram pruning (keep top *n* hypotheses in stack)**
- **Threshold pruning (keep hypotheses that are at most some factor of the best hypothesis in this stack)**

DECODING – SOME TECHNIQUES (3)

Motivation

Comparison among hypotheses requires future cost estimation

otherwise easy parts get preferred (organization according number of words)

Future cost estimation

Estimate costs for each translation option (consider remaining input)

- **Estimate language model costs (according to bi- or tri-grams)**
- **Find cheapest cost among translation options**
- **Ignore reordering costs**
- **Find cheapest cost path for each span**
- **Updated by dynamic programming**

STACK-BASED DECODING (A*)

Functionality

Generate hypotheses incrementally in a best-first fashion

Output best hypothesis if a complete sentence, otherwise choose next best

Properties of building hypotheses and maintaining stack

Building order from left to right, but input consumption in any order

Use of several stacks (1 stack for each subset of input words)

to ensure comparability, fairness across stacks (a simplificaftion)

Operations: Add (1:1), AddZfert (1:2), Extend (2:1), AddNull (1:0)

Search costs – AddZfert very expensive

Limiting the candidate words in the target language

Only if probability of the hypothesis increases (Δ p(t) > Δ p(a,s | t))

GREEDY DECODING

Functionality

Start out with random, approximate solution (e.g., most likely 1:1 mappings)

Incrementally improving it, until satisfactory result obtained

Operations

TranslateOneOrTwoWords (including null mappings)

TranslateAndInsert (insertion at position with highest alignment probability)

RemoveWordOfFertility0

SwapSegments (applicable to non-overlapping parts in the goal expression)

JoinWords (changes alignment of target words)

General enough to enable the decoder to escape local maxima

Relatively inexpensive

GREEDY DECODING – EXAMPLE (1)

NULL well heard , it talking a beautiful victory .

bien entendu , il parle de une belle victoire .

TranslateTwoWords(5,talks,7,great)

NULL well heard , it talks a great victory .

bien entendu , il parle de une belle victoire .

TranslateTwoWords(2,understood,0,about)

NULL well understood , it talks about a great victory .

bien entendu , il parle de une belle victoire .

GREEDY DECODING – EXAMPLE (2)

TranslateOneWord(4,he)

NULL well understood , he talks about a great victory .

bein entendu , il parle de une belle victoire .

TranslateTwoWords(1,quite,2,naturally)

NULL quite naturally , he talks about a great victory .

bein entendu , il parle de une belle victoire .

Altogether, the decoder explores a total of 77421 distinct alignments/translations

INTEGER PROGRAMMING DECODING (1)

Functionality

Mapping decoding onto a traveling salesman problem

(good word order is similar to a good TSP tour)

Expressing machine translation in integer programming format (à la TSP)

There is a city for each word in the source sentence

In each city, there are a number of hotels, corresponding to word translations

If two cities have hotels with same owner, add extra hotel on city borderline

(generalized to *n* cities in which same owner has hotels)

City tour is a sequence of hotels, exactly one from each city

Hotels on borderlines count for both cities

Each tour of cities corresponds to potential decoding

INTEGER PROGRAMMING DECODING (2)

Real-valued distances between pairs of hotels

-log(p(t)*p(a,s | t)) – converting language and translation probabilities

special treatment of "NULL hotels" (limited)

infinite costs between hotels within the same city

Example: 6 word sentence, about 80 hotels, and 3500 finite cost segments

Cast tour selection as an integer program – subtour elimination

Binary variable for each pair of hotels (it is or is not on the best path)

Objective function is the sum of all pairs weighted by distance costs

Constraint 1 – All cities visited exactly once: Sum of pair variables per city =1

Constraint 2 – Connected path: Sum of ingoing equals outgoing per hotel

Constraint 3 – Avoid sub-tours: Every proper subset of cities is visited

COMPARISON

Sentence length	*Decoder type*	*Time (sec)*	*Search errors*	*Translation errors*
6	**IP**	**47.50**	**0**	**57**
6	**stack**	**0.79**	**5**	**58**
6	**greedy**	**0.07**	**18**	**60**
6	**IP**	**499.00**	**0**	**76**
6	**stack**	**5.67**	**20**	**75**
6	**greedy**	**2.66**	**43**	**75**

Set of 101 test sentences

Experiments with bi-gram language model

(Germann et al. 2001)

PHRASAL TRANSLATION

Representing phrases

1-to-N mappings by fertility

Extending to N-to-N mappings

Mixed with word-to-word translations

Limiting phrasal length to force translations to be of similar length

Decoding

Tries to find most plausible parse tree in target language for source sentence

English context-free grammar extended by channel operations

(reordering, inserting, translating) and their associated probabilities

Best tree has highest probabilities in language model and in operations

Impractical when vocabulary and rule sizes grow

PRUNING IN PHRASAL TRANSLATION

Beam search

Dynamic programming parser – costs outside subtree neglected

Translation operation pruning

Only the top n word-to-word translations are considered (top 5 used)

Phrase pruning

Pairs must appear more than once in Viterbi alignments from training corpus

Reordering rules pruning (875 out of 138,662)

Only consider top-ranked rules such that the effect of them takes care of > 95%

Limiting target word insertion

Top 20 target words and top 20 contexts considered, cover over 60%

A SHIFT-REDUCE DECODING ALGORITHM

Conceptual idea

Inversion Transduction Grammars (ITGs) permit ordering flexibility

Well-suited for modeling ordering shifts between languages

Left-to-right processing allows accurate language model calculations

Differences to ITG parsing

No ambiguities considered so far

Reorderings must fulfil ITG constraints

General idea

Phrases in source language conceived as blocks

Adjacent blocks can be merged irrespective of ordering

Derivation structure does not affect scoring (reordering probabilities)

THE ALGORITHM

Correspondences between source words and target phrases

Potetially much reordering between Chinese and English

Adjacent blocks can be merged (in the example):

A_2 matches the block [zairu$_1$] and **yield**

A_8 matches the block [shij ian$_2$ diaocha$_3$ ziliaode$_4$] **block A_9**

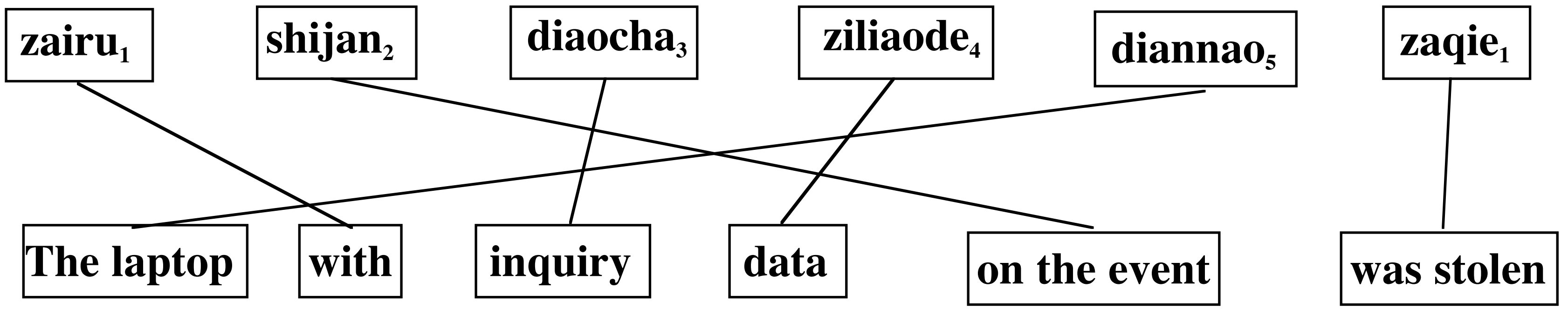
Constraints to choose next block to cover

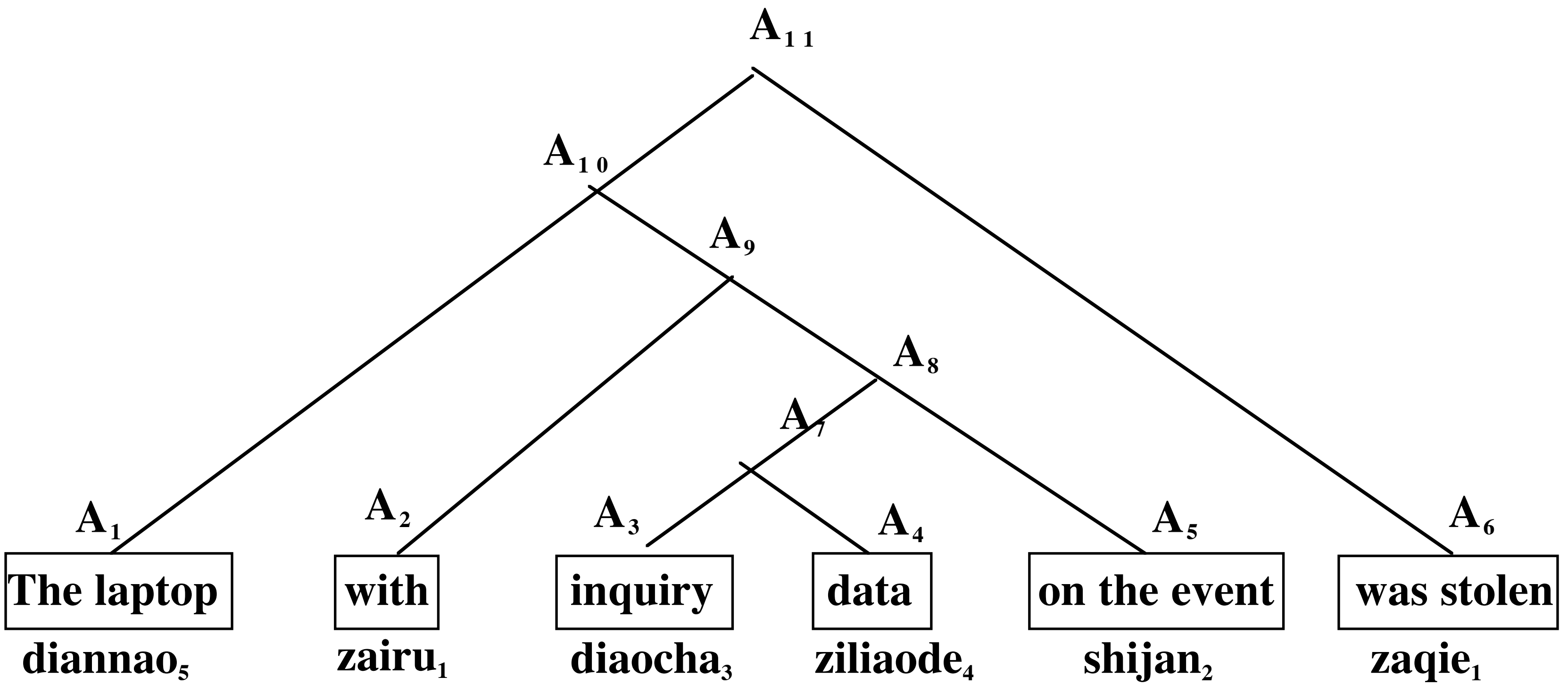
Left/right word covered and right/left word uncovered

-> choose next block from right/left uncovered span

Choices applied recursively, until all words merged into a single block

It can be shown that (only) these choices ensure obeying ITG constraints


zairu$_1$
shijan$_2$
diaocha$_3$
ziliaode$_4$
diannao$_5$
zaqie$_1$
The laptop
with
inquiry
data
on the event
was stolen


A_{11}
A_{10}
A_9
A_8
A_7
A_1
A_2
A_3
A_4
A_5
A_6
The laptop
with
inquiry
data
on the event
was stolen
diannao$_5$
zairu$_1$
diaocha$_3$
ziliaode$_4$
shijan$_2$
zaqie$_1$

THE ALGORITHM – OPERATIONS

Data structures / states

A stack of covered blocks (in the order in which they are covered)

A stack for the left uncovered spans of the current block

A stack for the right uncovered spans of the current block

3 basic operations (informally)

***Lshift* pop top element of stsck of covered blocks,**

choose one block from the left neighborhood, if adjacent word not covered

and update all three stacks

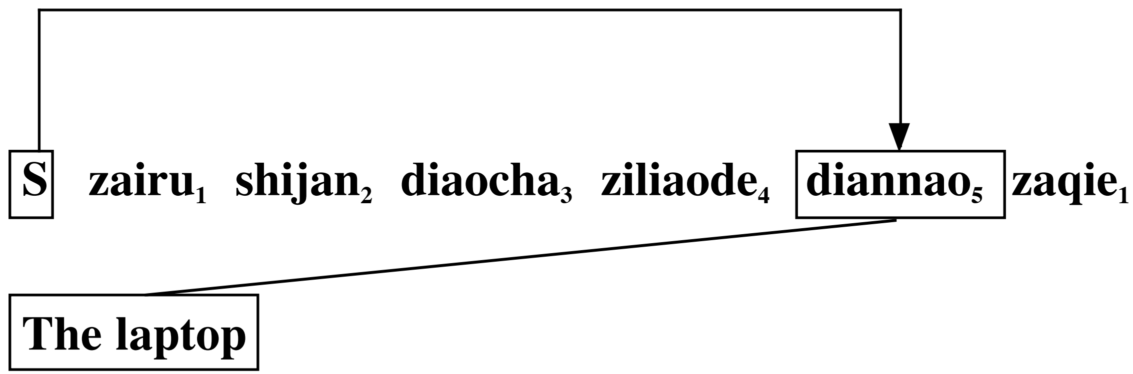
***Rshift* analoguously**

Reduce pop the two top blocks and push the merged one

if these two blocks are adjacent

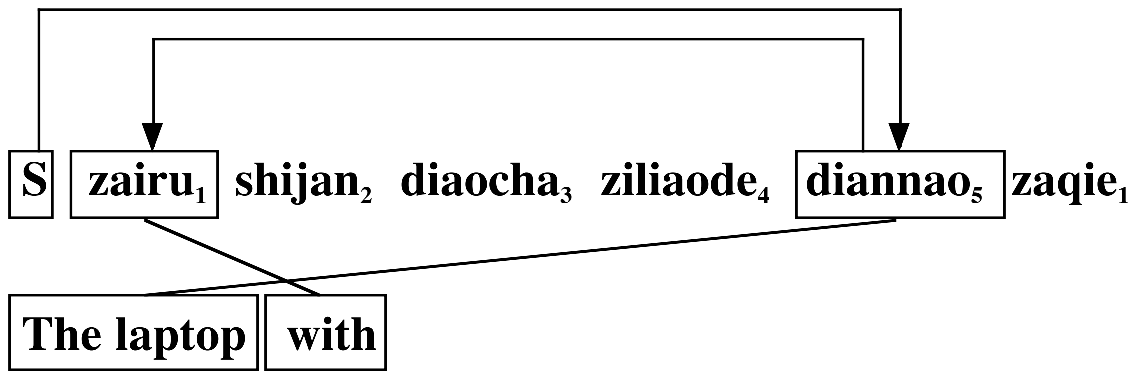
THE ALGORITHM – AN EXAMPLE (1)

Operation	*Stack top blocks*	*Stack left*	*Stack right*
RShift	**[0][5]**	**[1, 4]**	**[6]**



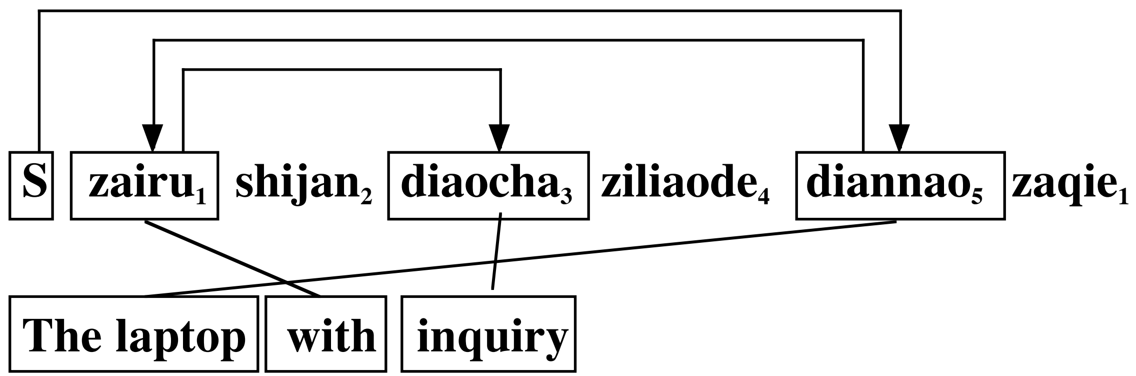
THE ALGORITHM – AN EXAMPLE (2)

Operation	*Stack top blocks*	*Stack left*	*Stack right*
LShift	**[0][5] [1]**	**Ø**	**[2,4] [6]**



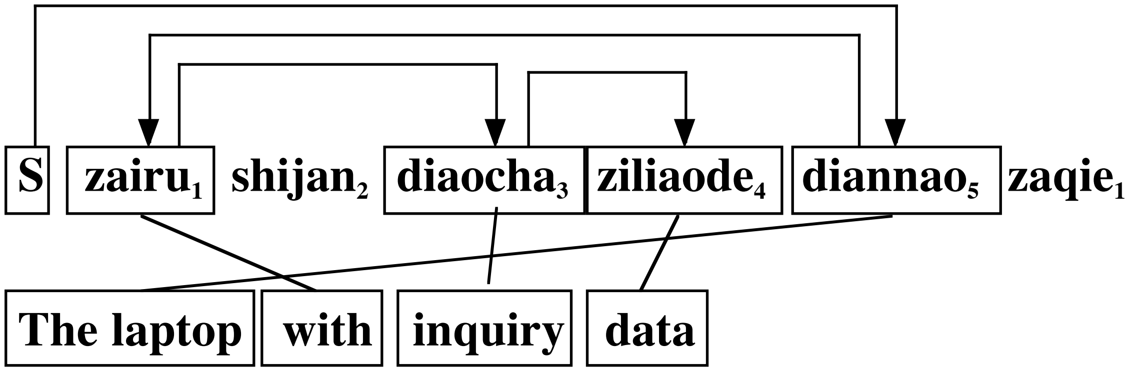
THE ALGORITHM – AN EXAMPLE (3)

Operation	*Stack top blocks*	*Stack left*	*Stack right*
RShift	**[0][5][1][3]**	**[2]**	**[4][6]**



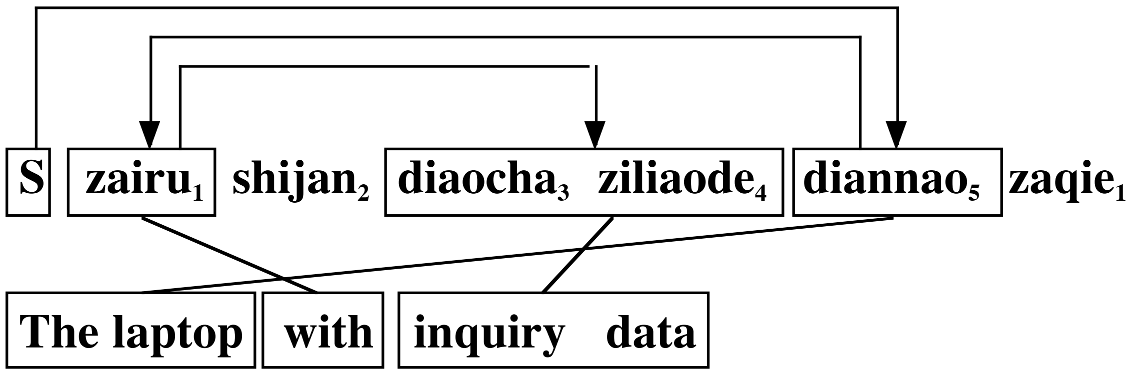
THE ALGORITHM – AN EXAMPLE (4)

Operation	*Stack top blocks*	*Stack left*	*Stack right*
RShift	**[0][5] [1][3][4]**	**[2]**	**[6]**



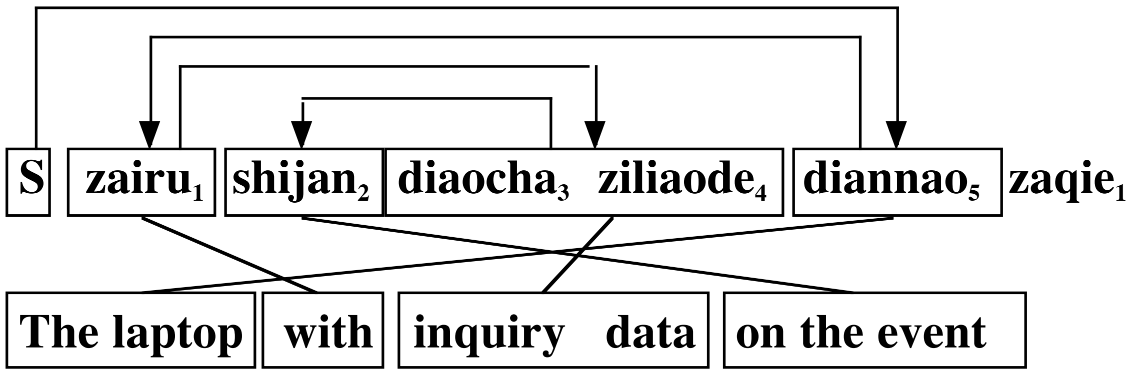
THE ALGORITHM – AN EXAMPLE (5)

Operation	*Stack top blocks*	*Stack left*	*Stack right*
Reduce	**[0][5][1][3,4]**	**[2]**	**[6]**



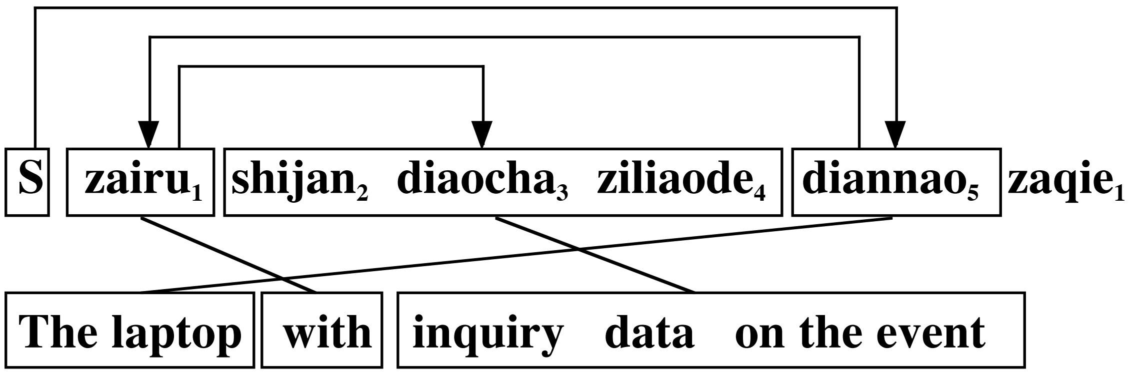
THE ALGORITHM – AN EXAMPLE (6)

Operation	*Stack top blocks*	*Stack left*	*Stack right*
LShift	**[0][5][1][3,4][2]**	**Ø**	**[6]**



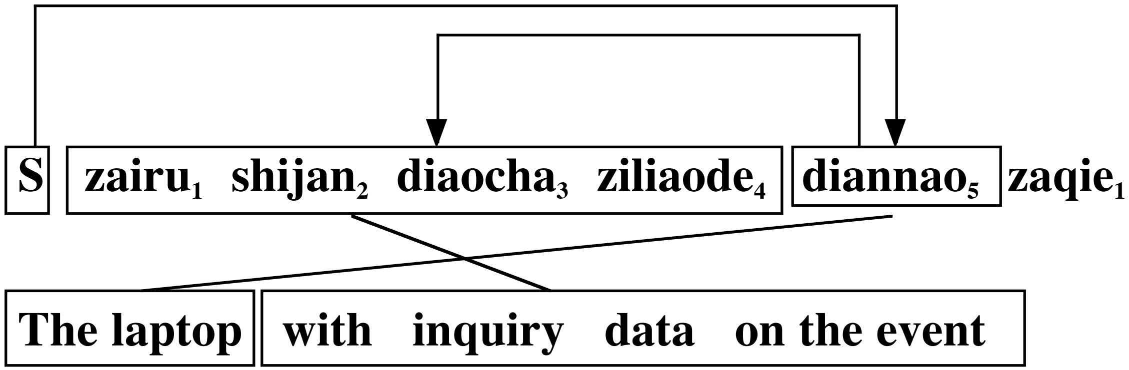
THE ALGORITHM – AN EXAMPLE (7)

Operation	*Stack top blocks*	*Stack left*	*Stack right*
Reduce	**[0][5] [1][2,4]**	**Ø**	**[6]**



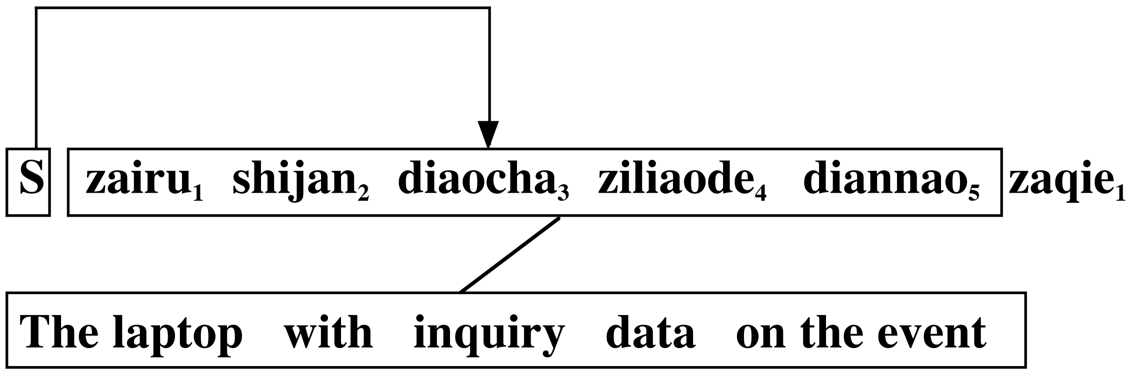
THE ALGORITHM – AN EXAMPLE (8)

Operation	*Stack top blocks*	*Stack left*	*Stack right*
Reduce	**[0][5] [1,4]**	**Ø**	**[6]**



THE ALGORITHM – AN EXAMPLE (9)

Operation	*Stack top blocks*	*Stack left*	*Stack right*
Reduce	**[0][1,5]**	**Ø**	**[6]**



SHIFT-REDUCE DECODING ASSESSMENT

Experiments

Comparisons with decoder MOSES

Improved accuracy and speed with (informed) ITG constraints

Competitive in other settings

Rationale

Performs reduce operations as early as possible

Can exploit constraining relations very effectively

Assessment

How easy/reliable is the derivation of ITG constraints

Extensions to ambiguities – parallel processing?

Authors intend to apply the technique to syntax-based statistical MT

PHARAOH – PHRASE-BASED DECODER

Language model

Additional feature functions (reordering, word penalty, phrase translation)

Search techniques

All phrase translations options computed a priori

Hypothesis regeneration applied

Beam search with non-admissible functions (threshold and histogram pruning)

Experiments

Trained on 30 Million word German-English corpus (European Parliament)

Test set 1500 sentences, average length 28.9 words (German to English)

Best results for 0.1 threshold, maximum stack size 1000, translation table 50

PHARAOH – EXPERIMENTAL RESULTS

Threshold pruning

Threshold	0.0001	0.001	0.01	0.05	0.08	0.1	0.15	0.2	0.3
Time	149 sec	119 sec	70 sec	27 sec	18 sec	15 sec	13 sec	10 sec	7 sec
Search error	-	+0%	+0%	+0%	+0%	+1%	+3%	+4%	+12%

Histogram pruning

Beam size	1000	200	100	50	20	10	5
Time	15 sec	15 sec	14 sec	10 sec	9 sec	9 sec	7 sec
Search error	+1%	+1%	+2%	+4%	+8%	+20%	+35%

Translation table size

T-Table limit	1000	500	200	100	50	20	10	5
Time	15 sec	7.6 sec	3.8 sec	1.9 sec	0.9 sec	0.4 sec	0.2 sec	0.1 sec
Search error	+1%	+1%	+1%	+1%	+1%	+2%	+7%	+18%

MACHINE TRANSLATION WITH A* (Och, Ueffing, Ney)

Motivation

Alternative to beam search

“Observation” pruning – top 12 translations of at least 1% probability

Search much more efficient and optimality obtained within restricted repertoire

Techniques and results

Optimal translation costs precomputed for remaining words

Heuristic function comprising language and translation models and fertility

Heuristic function for reordering depends on the model used

Compromise – almost admissible heuristic function, big savings

Translation quality competes with beam search, but is much slower

SYNTAX-BASED TRANSLATION MODELS

Stochastic operations on parse tree

Reordering of nodes on the same hierarchical level

Inserting optional extra words

Translating leaf words

Probabilities of this channel conditioned by the previous context

Motivation and drawback

Syntax (e.g., long distance dependencies) badly captured in word-based models

Can guarantee the production of sentences (e.g., when translating Chinese)

Syntax-based model can be superior

Problems with learning good language models

Sparse data

A SYNTAX-BASED TRANSLATION MODEL (BBN)

Conceptual approach

String-to-dependency (extends hierarchical string-to-string)

Independent of source parse quality

Fixed and floating (partial) dependepcy structures

Combination through adjoinment and concatenation

Rule extraction

Based on word alignment and target dependency

Rule set enhanced through inferences (generalizations, limited by complexity

Search methods

Chart parsing, hypotheses organized in AND-OR-structures

Beam search used to cut down the complexity

COMPARISON BETWEEN SYNTAX-BASED AND PHRASE-BASED TRANSLATION MODELS

Language model learning

Overlap between models learned approx. 50%

(due to various constraints, limits, …)

But also unaligned words in syntactically motivated locations

(Potential) coverage

Syntax-based models attributed with higher potential for improvements

Strength of the phrase-based extraction model to be exploited

Performance

Phrase-based models are generally superior (until recently)

Syntax-based models may be problematic in their decoder efficiency

RECENT MACHINE TRANSLATION ARCHITECTURES

Two-pass search strategy

1. Decoding to generate N-best list of translation hypotheses

2. Final translation selected by rescoring and re-ranking

Motivation

Global sophisticated and discriminative feature functions

Some functions cannot easily be decomposed

Extensions

Hierarchical models (phrases with gaps, to be filled by subphrases)

System combinations – N-best hypotheses generated by several systems

Regeneration (intermed. step): recoding, N-gram expansion, confusion network

"it's about 5 minutes"	**+**	**"5 minutes on"**	**->**	**"it's about 5 minutes on"**
(partial hypothesis)		**(N-gram)**		**(extended hypothesis)**