

TectoMT

Software framework for developing
MT systems (and other NLP applications)

Zdeněk Žabokrtský
ÚFAL MFF UK

Outline



- Part I - Introduction
 - What is TectoMT
 - Motivation
- Part II - TectoMT System Architecture
 - Data structures
 - Processing units: blocks, scenarios, applications
- Part III - Applications implemented in TectoMT

What is TectoMT

- TectoMT is ...
 - a highly modular extendable NLP software system
 - composed of numerous (mostly previously existing) NLP tools integrated into a uniform infrastructure
 - aimed at (not limited to) developing MT system
- TectoMT is not ...
 - a specific method of MT (even if some approaches can profit from its existence more than others)
 - an end-user application (even if releasing of single-purpose stand-alone applications is possible and technically supported)

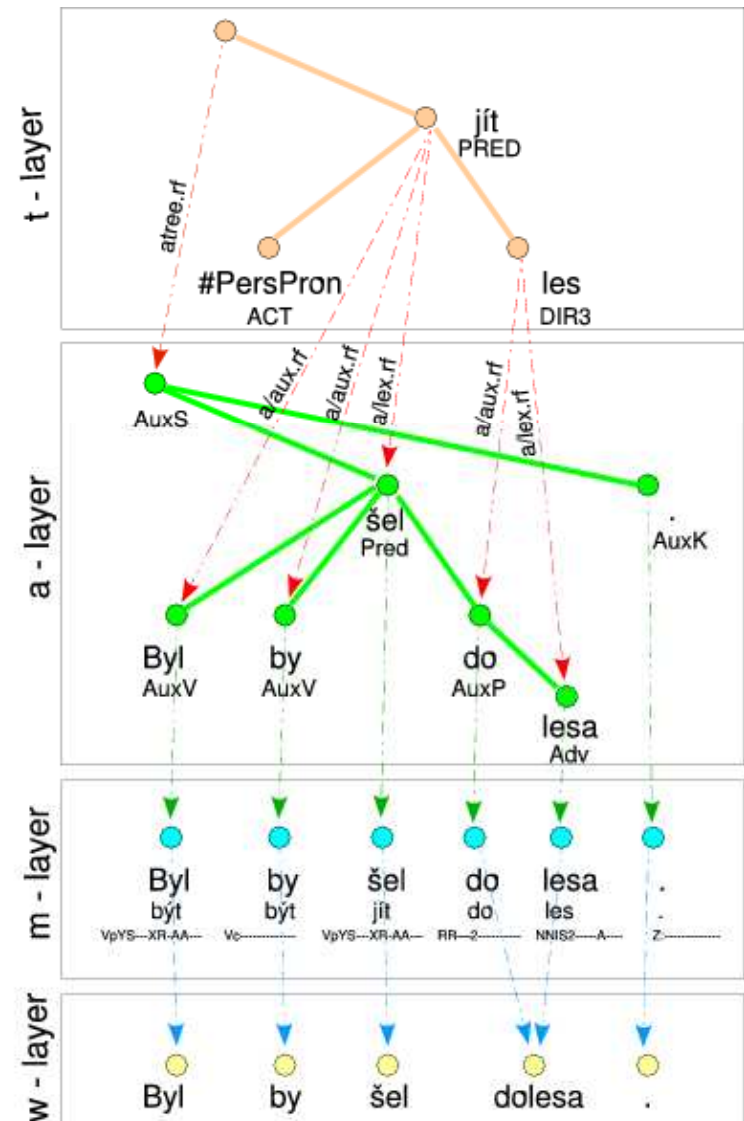
Motivation for creating TectoMT

- First, technical reasons:
 - Want to make use of more than two NLP tools in your experiment? Be ready for endless data conversions, need for other people's source code tweaking, incompatibility of source code and model versions...
 - Unified software infrastructure might help us.
- Second, our long-term MT plan:
 - We believe that tectogrammar (deep syntax) as implemented in Prague Dependency Treebank might help to (1) **reduce data sparseness**, and (2) find and **employ structural similarities** revealed by tectogrammar even between typologically different languages.

Prague Dependency Treebank 2.0

- three layers of annotation:
 - tectogrammatical layer
 - deep-syntactic dependency tree
 - analytical layer
 - surface-syntactic dependency tree
 - 1 word (or punct.) ~ 1 node
 - morphological layer
 - sequence of tokens with their lemmas and morphological tags

[Ex: *He would have gone into forest*]

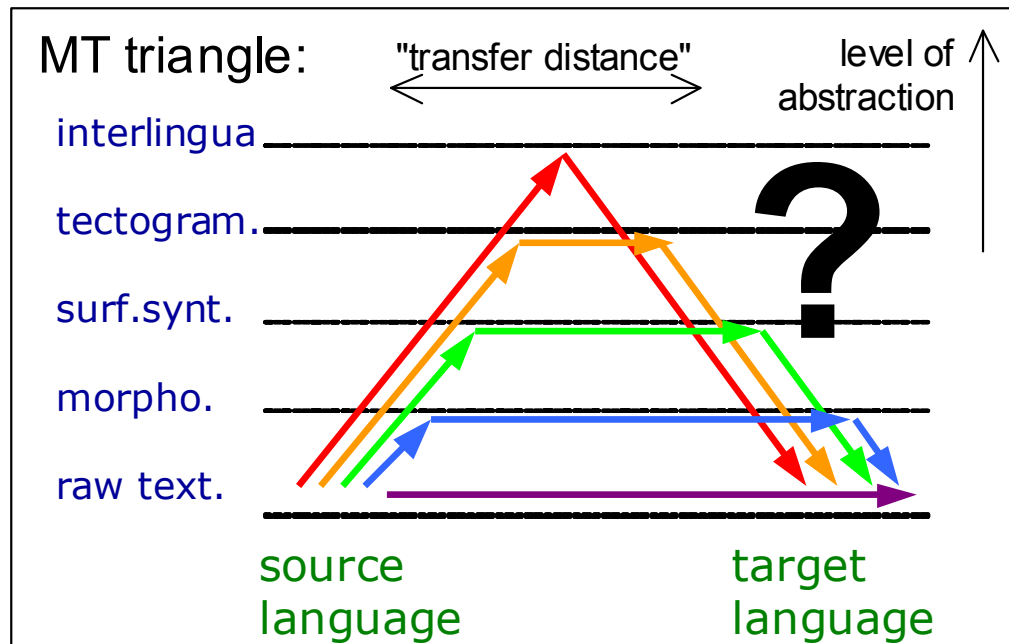


Tectogrammar in a nutshell

- tectogrammatical layer of language representation
 - introduced by Petr Sgall in 1960's, implemented in PDT 2.0
- key features:
 - each sentence represented as a **deep-syntactic dependency tree**
 - **functional words** (such as aux.verb, prepositions, subordinating conjunctions) accompanying an autosemantic word "**collapse**" with it into a single t-node, labeled with the autosemantic t-lemma
 - "added" nodes (e.g. because of pro-dropped subjects)
 - semantically indispensable syntactic and morphological knowledge represented as attributes of nodes
 - **economy**: no nonterminals, less nodes than words in the original sentence, decreased morphological redundancy (categories imposed by agreement disappear), etc.

MT triangle in terms of PDT

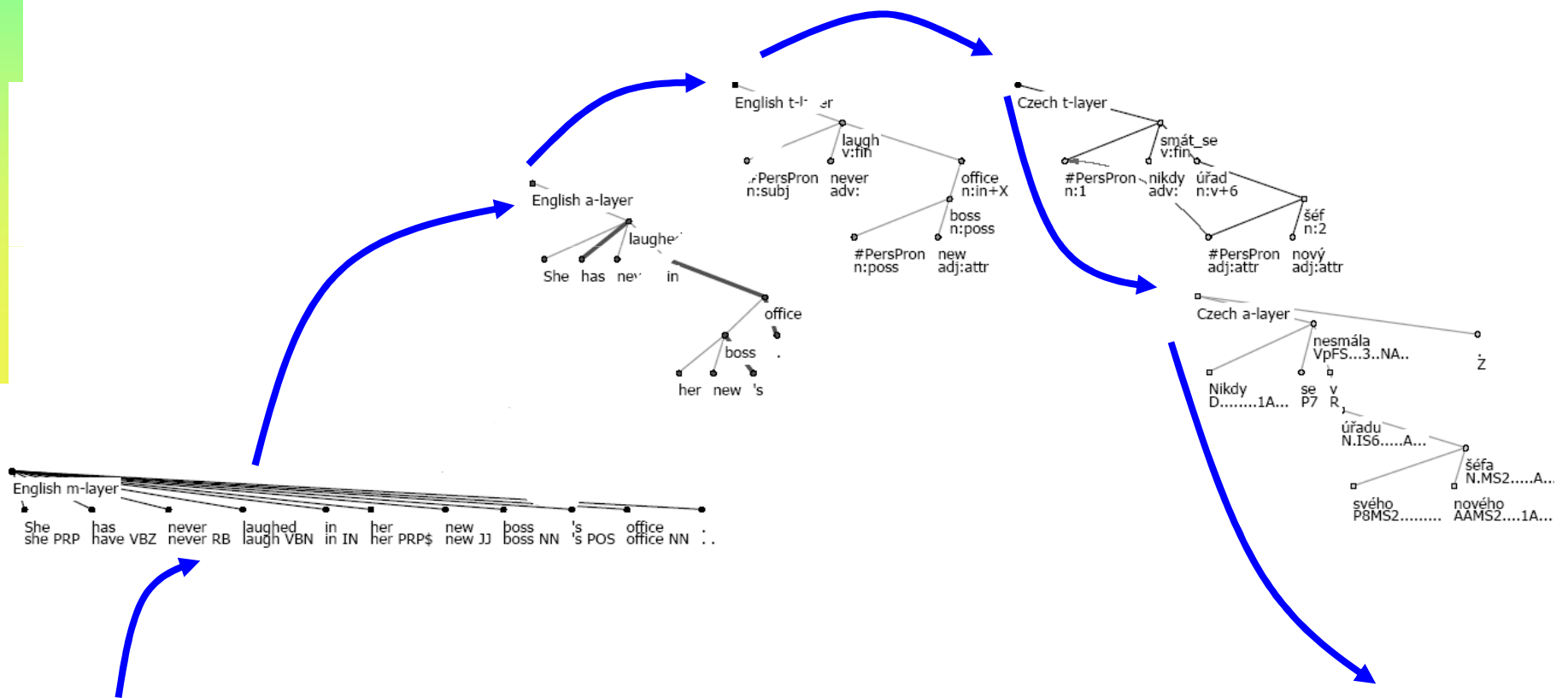
- Key question: what is the optimal level of abstraction?



- Obvious trade-off: ease of transfer vs. additional analysis and synthesis costs (system complexity, errors...)

MT triangle *in vivo*

- Illustration: analysis-transfer-synthesis in TectoMT



She has never laughed in her new boss's office.

Nikdy se nesmála v úřadu svého nového šéfa.

How could tecto help?

- Vague assumption:
 - tectogrammatics **abstracts from several language-specific characteristics** (e.g. makes no difference between meanings expressed by isolated words, inflection or agglutination)
 - ...therefore **languages look more similar** at the tecto-layer
 - ...therefore the **transfer phase should be easier** (compared to the operation on raw sequences of word forms)
- Yes, but how exactly could it help?

How could tecto help? (cont.)

■ n-gram view:

- manifestations of lexemes are mixed with manifestations of language means expressing the relations between the lexemes and of other grammar rules
 - inflectional endings, agglutinative affixes, functional words, word order, punctuation orthographic rules ...
 - *It will be delivered to Mr. Green's assistants at the nearest meeting.*
- → **training data sparsity**

■ tectogrammar view:

- clear separation of meaningful "signs" from "signs" which are only imposed by grammar (e.g. imposed by agreement)
- clear separation of lexical, syntactical and morphological meaning components
- → modularization of the translation task → **potential for a better structuring of statistical models** → more effective exploitation of the (limited) training data

Tecto transfer factorization

- Three transfer “channels” can be separated:
 - translation of lexicalization
 - E.g. 'koupit' goes to 'buy'
 - translation of syntactization
 - e.g. relative clause goes to attributive adjective
 - Translation of morphological meanings
 - e.g. singular goes to singular
- The channels are relatively loosely coupled (esp. the third one) which could be used for smoothing.

Tecto transfer factorization (cont.)

- Example: three ways to express future tense in Czech
 - (1) aux.verb: *budu ... chodit* - *I will walk ...*
 - (2) prefix: *poletím* - *I will fly ...*
 - (3) ending: *uvařím* - *I will boil ...*
- nontrivial tense translation from the n-gram view
- but once we work with tecto analyses, we can translate the future tense just to future tense, separately from translating the lemma
 - similarly, plural goes mostly to plural, comparative to comparative, etc.

Tecto transfer factorization (cont.)

- we introduce the notion of **formemes** - morphosyntactic language means expressing the dependency relation
- example values:
 - **n:v+6** (in Czech) = semantic noun which is on the surface expressed in the form of prepositional group in locative with preposition "v"
 - **v:that+fin/a** (in English) = semantic verb expressed in active voice as a head of subordinating clause introduced with the sub.conjunction "that"
 - **v:rc** (in Czech and English) = head of relative clause
 - **n:sb** (in English) = noun in subject position
 - **n:1** (in Czech) = noun in nominative case
 - **adj:attr** (in Czech and English) = adjective in attributive position
- formemes allow us to introduce a **separate syntactization factor** and to train it using a parsed parallel corpus

- trained estimates of $P(F_{cz} | P_{en})$:

v:to+inf	v:inf	0.4817
v:to+inf	v:aby+fin	0.0950
v:to+inf	n:k+3	0.0702
v:to+inf	v:že+fin	0.0621
n:for+X	n:pro+4	0.2234
n:for+X	n:2	0.1669
n:for+X	n:4	0.0788
n:for+X	n:za+4	0.0775

Using tree context

- Hypothesis: translation choices are conditioned rather by governing/dependent words than by linear predecessors/followers
- syntactic dependency and linear adjacency often coincide, but long distance dependencies occur too
- long distance dependencies are notoriously difficult to handle by n-gram models

Using tree context (cont.)

■ Example 1:

- *The **grass** around your house should be **cut** soon.*
- google trans.: ***Trávu** kolem vašeho domu by se měl **snížit** v nejbližší době.*
- incorrect morphological choice with the subject; verb form is crucial for the correct choice, but it is too far
- incorrect lexical choice of the verb; subject's lexical occupation could help, but it is too far

■ Example 2

- *Zítra se v kostele Svaté Trojice budou **brát** Marie a Honza.*
- google trans: *Tomorrow is the Holy Trinity church will **take** Mary and John.*
- Incorrect lexical choice: presence of the "se" clitic at the clause-second position is crucial, but it is too far

How could tecto help - summary



- Tectogrammar offers a natural **transfer factorization** into three relatively independent channels
- Tectogrammar offers **local tree context** (instead of only local linear context)

Hybrid MT with TectoMT

- other option: to combine translation based on tecto-transfer with a conventional phrase-based translation system X
- TectoMT can **provide** X with **additional hypotheses**
- TectoMT can be used for **decomposing input sentences** into smaller, relatively independently translatable chunks (e.g. finite clauses or even individual constituents)
- TectoMT can **steal** the lexical choices from X's output and **resynthesize** the sentence (or its parts) according to grammar rules, e.g. in order to correct agreement
- New features for reranking X's output hypotheses can be extracted from their syntactic analyses by TectoMT (e.g. by penalizing presence of abnormal tree configurations)

Hybrid MT with TectoMT (cont.)

- TectoMT can be used for making the source and target languages more similar even from the n-gram view:
 - Adding artificial tokens (e.g. inserting `_det_` when translating to a language with determiners)
 - Joining tokens (e.g. `of John` -> `of_John`, when translating into a language using genitive ending instead of a functional word)
 - Regular grammar-based word order changes: e.g. shifting *ago* in front of the noun group (as it was a preposition) when translating from English to German

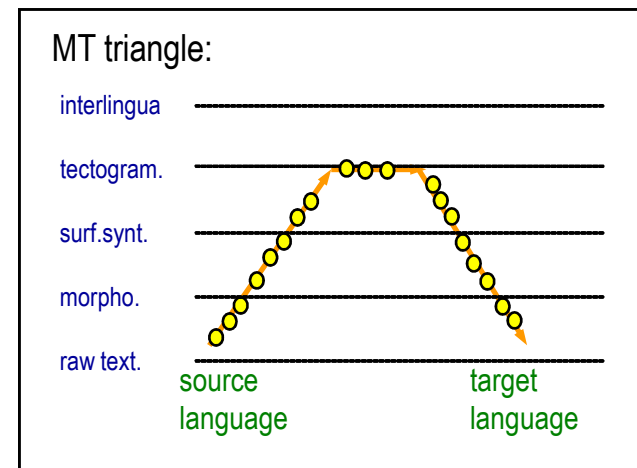


Part II:

TectoMT System Architecture

Design decisions

- **Linux + Perl**
- set of well-defined, **linguistically relevant layers** of language representation
- **neutral** w.r.t. chosen methodology ("rules vs. statistics")
- accent on modularity: translation **scenario** as a sequence of translation **blocks** (modules corresponding to individual NLP subtasks)
 - reusability
 - substitutability

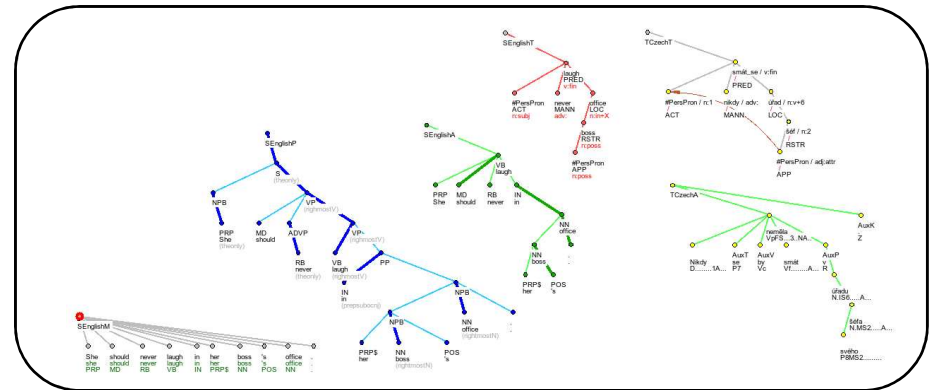


Design decisions (cont.)

- reuse of Prague Dependency Treebank technology (tools, XML-based format)
- in-house **object-oriented architecture** as the backbone
 - all tools communicate via standardized OO Perl interface
 - avoiding the former practice of tools communicating via files in specialized formats
- **easy incorporation of external tools**
 - previously existing parsers, taggers, lemmatizers etc.
 - just provide them with a Perl "wrapper" with the prescribed interface

Hierarchy of data-structure units

- **document**
 - the smallest independently storable unit (~ xml file)
 - represents a text as a sequence of bundles, each representing one sentence (or sentence tuples in the case of parallel documents)
- **bundle**
 - set of tree representations of a given sentence
- **tree**
 - representation of a sentence on a given layer of linguistic description
- **node**
- **attribute**
 - document's, node's, or bundle's attrname-value pair



Layers of sentence description

- in each bundle, there can be at most one tree for each "layer"
- set of possible layers = $\{S,T\} \times \{\text{English,Czech,...}\} \times \{M,P,A,T,N\}$
 - S - source, T-target
 - M - morphological analysis
 - P - phrase-structure tree
 - A - analytical tree
 - T - tectogrammatical tree
 - N - instances of named entities
- Example: SEnglishA - tectogrammatical analysis of an English sentence on the source-language side

Hierarchy of processing units

■ block

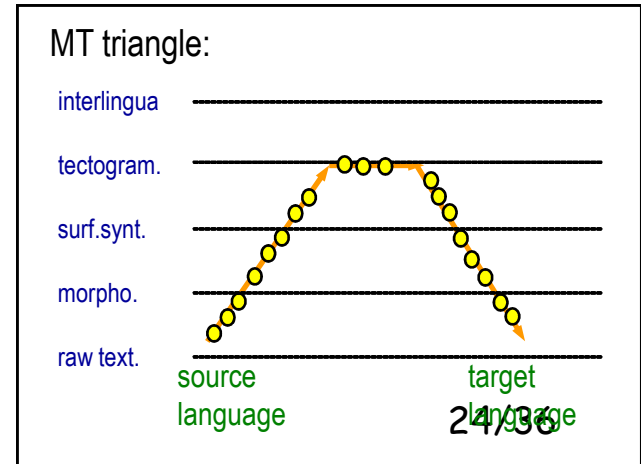
- the smallest individually executable unit
- with well-defined input and output
- block parametrization possible (e.g. model size choice)

■ scenario

- sequence of blocks, applied one after another on given documents

■ application

- typically 3 steps:
 - 1. conversion from the input format
 - 2. applying the scenario on the data
 - 3. conversion into the output format



Blocks

- technically, Perl classes derived from TectoMT::Block
- either method `process_bundle` (if sentences are processed independently) or method `process_document` must be defined
- more than 200 blocks in TectoMT now, for various purposes:
 - blocks for analysis/transfer/synthesis, e.g.
 - `SEnglishW_to_SEnglishM::Lemmatize_mtree`
 - `SEnglishP_to_SEnglishA::Mark_heads`
 - `TCzechT_to_TCzechA::Vocalize_prepositions`
 - blocks for alignment, evaluation, feature extraction, etc.
- some of them only implement simple rules, some of them call complex probabilistic tools
- English-Czech tecto-based translation currently composes of roughly 80 blocks

Tools integrated as blocks


- to integrate a stand-alone NLP tool into TectoMT means to create a block that encapsulates the functionality of the tool behind the standardized block interface
- already integrated tools:
 - taggers
 - Hajič's tagger, Raab&Spoustová Morče tagger, Rathnaparkhi MXPOST tagger, Brants's TnT tager, Schmid's Tree tagger, Coburn's Lingua::EN::Tagger
 - parsers
 - Collins' phrase structure parser, McDonalds dependency parser, ZŽ's dependency parser
 - named-entity recognizer
 - Stanford Named Entity Recognizer, Kravalová's SVM-based NE recognizer
 - several other
 - Klimeš's semantic role labeller, ZŽ's C5-based afun labeller, Ptáček's C5-based Czech preposition vocalizer, ...

Other TectoMT components

- "core" - Perl libraries forming the core of TectoMT infrastructure, esp. for memory representation of (and interface to) to the data structures
- numerous file-format converters (e.g. from PDT, Penn treebank, Czeng corpus, WMT shared task data etc. to our xml format)
- TectoMT-customized Pajas' tree editor TrEd
- tools for parallelized processing (Bojar)
- data, esp. trained models for the individual tools, morphological dictionaries, probabilistic translation dictionaries...
- tools for testing (regular daily tests), documentation...

TectoMT directory structure

- everything under one directory tree specified in system variable TMT_ROOT
- **versioned part (in a svn repo)**
 - install/
 - libs/{core,blocks,packaged,other}/
 - tools/
 - applications/
 - doc/
 - personal/
 - tools/
 - training/
 - release_building/
 - evaluation/
- **shared part (unversioned)**
 - share/installed_tools/
 - share/installed_libs/
 - share/data/{models,resources...}
 - share/tred/

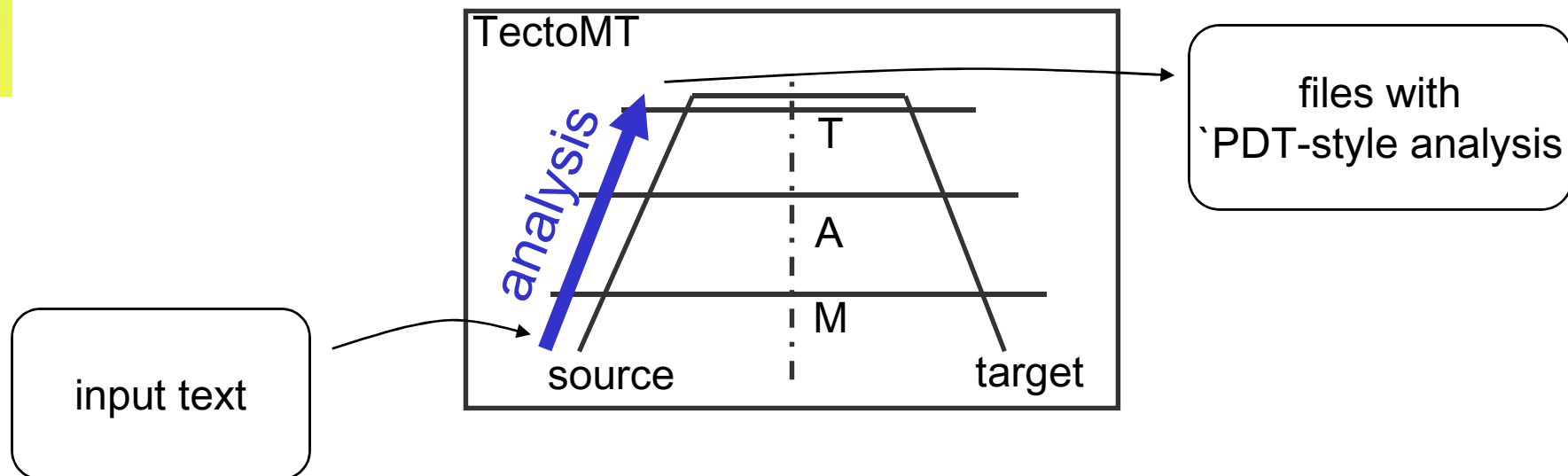


Part III:

Applications
implemented in TectoMT

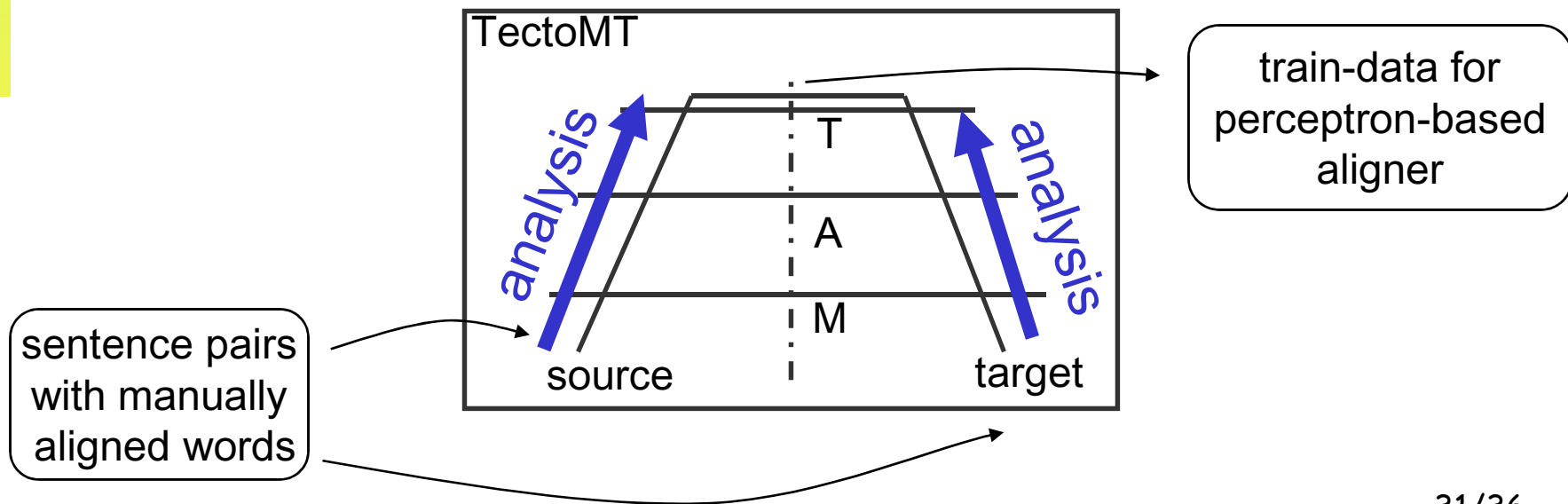
PDT-style layered analysis

- analyze a given Czech or English text up to morphological, analytical and tectogrammatical layer
- used currently e.g. in experiments with intonation generation or information extraction



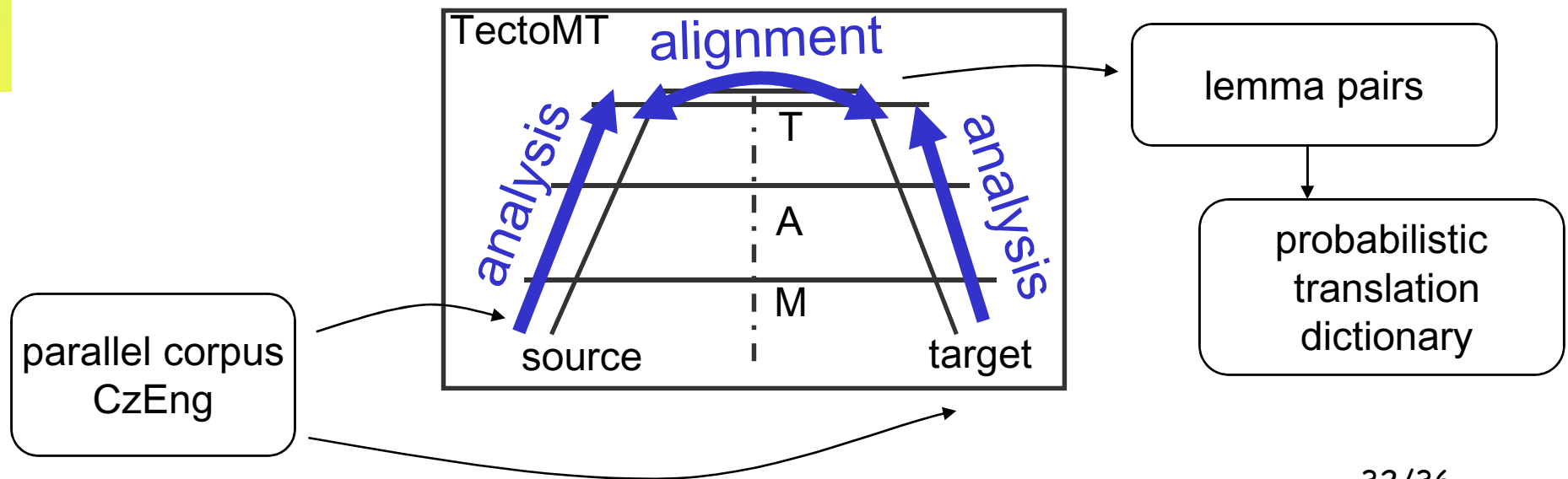
Training tecto-aligner

- data for training a perceptron-based aligner of tectogrammatical nodes, using manually sentence pairs aligned at the word layer
- the resulting aligner was used for aligning CzEng (parsed Czech-English parallel corpus, around 60MW)



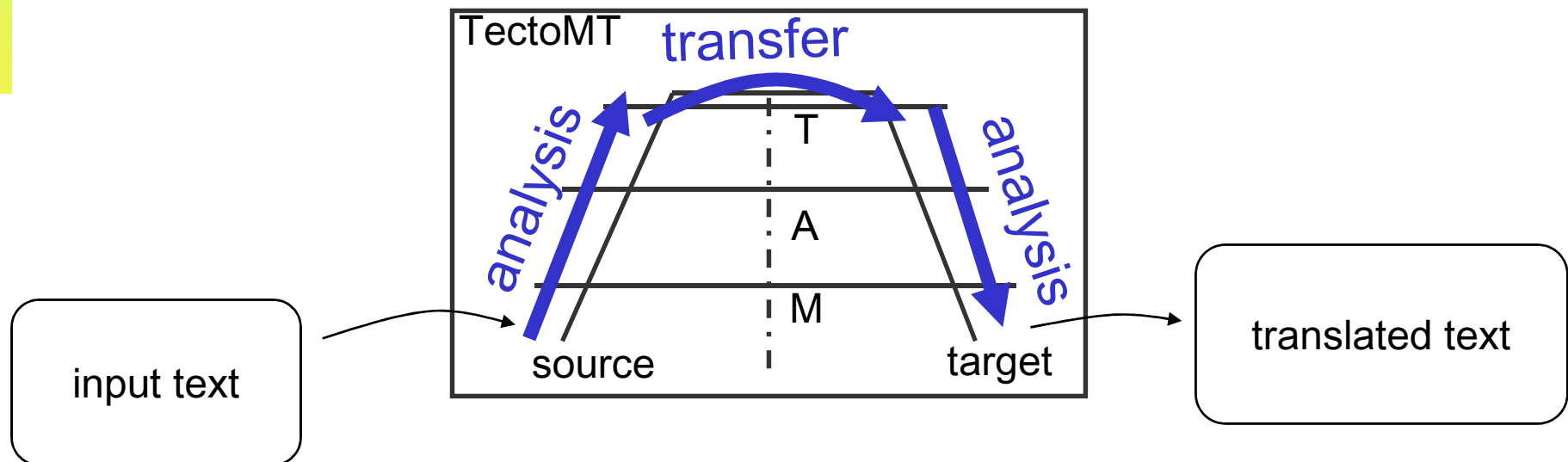
Transl. dictionary extraction

- using the lemma pairs from the aligned t-nodes from a huge parallel corpus, we build a probabilistic translation dictionary



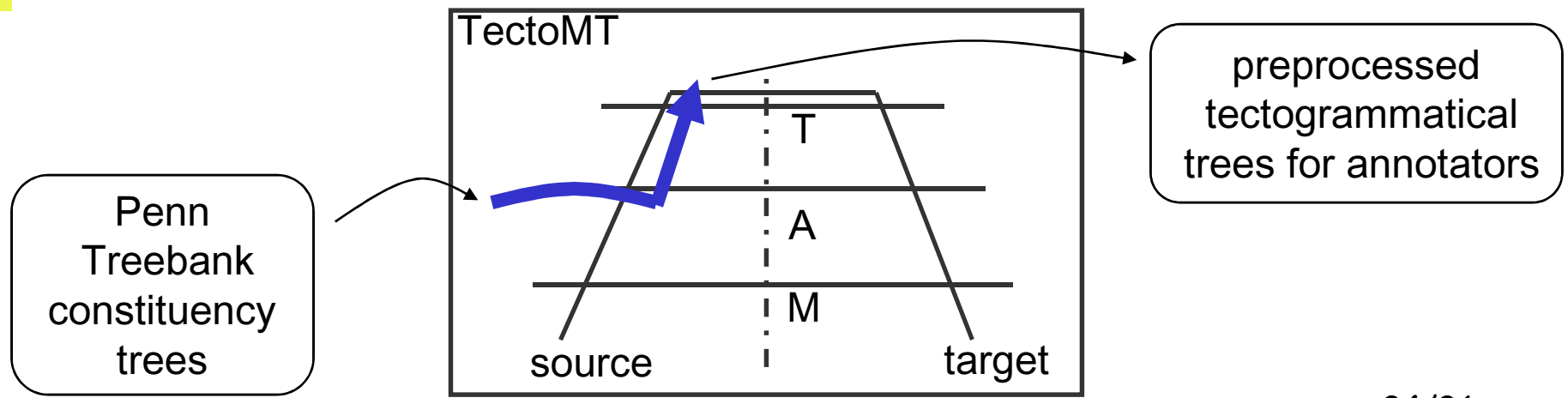
Translation with tecto-transfer

- analysis-transfer-synthesis translation from English to Czech and vice versa
- employed probabilistic dictionary from the previous slide



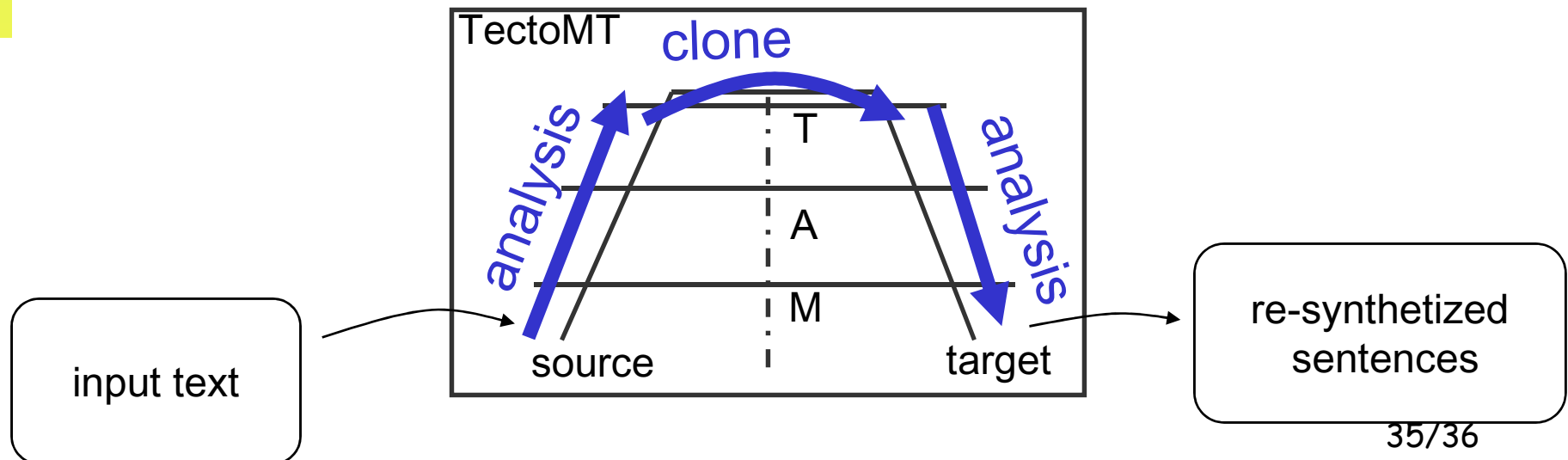
Preproc. data for PEDT

- Prague English Dependency Treebank
 - PDT-style annotation project at UFAL
 - currently 12000 English tectogrammatically analyzed sentences, since 2006, now 6 annotators, <http://ufal.mff.cuni.cz/pedt>
 - saving annotators' work by automatizing a part of the analysis in TectoMT



Sentence re-synthesis

- analysis-clone-synthesis scenario for
 - **postprocessing of other MT system's output** (to make it more grammatical)
 - **speech reconstruction** - postprocessing of STT's output (to make it more grammatical)
 - (useful also finding bugs anywhere along the scenario)
- very preliminary stage



Final remarks

- Our implementation of tectogrammar-based MT is still premature and does not reach state-of-the-art quality (WMT Shared Task 2009)
- However, having the TectoMT infrastructure and sharing its components already saves our work in several research directions.