Rudolf Rosa and David Mareček:

Dependency Relations Labeller for Czech

Dependency Relation Labelling

- assign a label from a given set of labels to each edge between two nodes (a child node and its parent node) in a dependency tree
- the label qualifies the type of relation between the two nodes, such as an adjective being an attribute of a noun, or a noun being a subject to a verb
- Czech language: 27 labels defined by the *Prague Dependency Treebank*: Predicate, Subject, Object, Attribute, Adverbial, Preposition...

Labelling Method

- as described in Ryan McDonald: *Discriminative learning and spanning tree algorithms for dependency parsing* (2006)
- independent second stage to unlabelled dependency parsing
- can be used as a second stage to any parser, even labelled
- tree processing direction: top-down, left-to-right (makes use of already made decisions)
- feature-based score factorisation:

  score (edge, label) = Σ score (feature, label)
- 68 feature templates based on 21 feature functions
- training: Margin Infused Relaxed Algorithm (MIRA)

Margin Infused Relaxed Algorithm

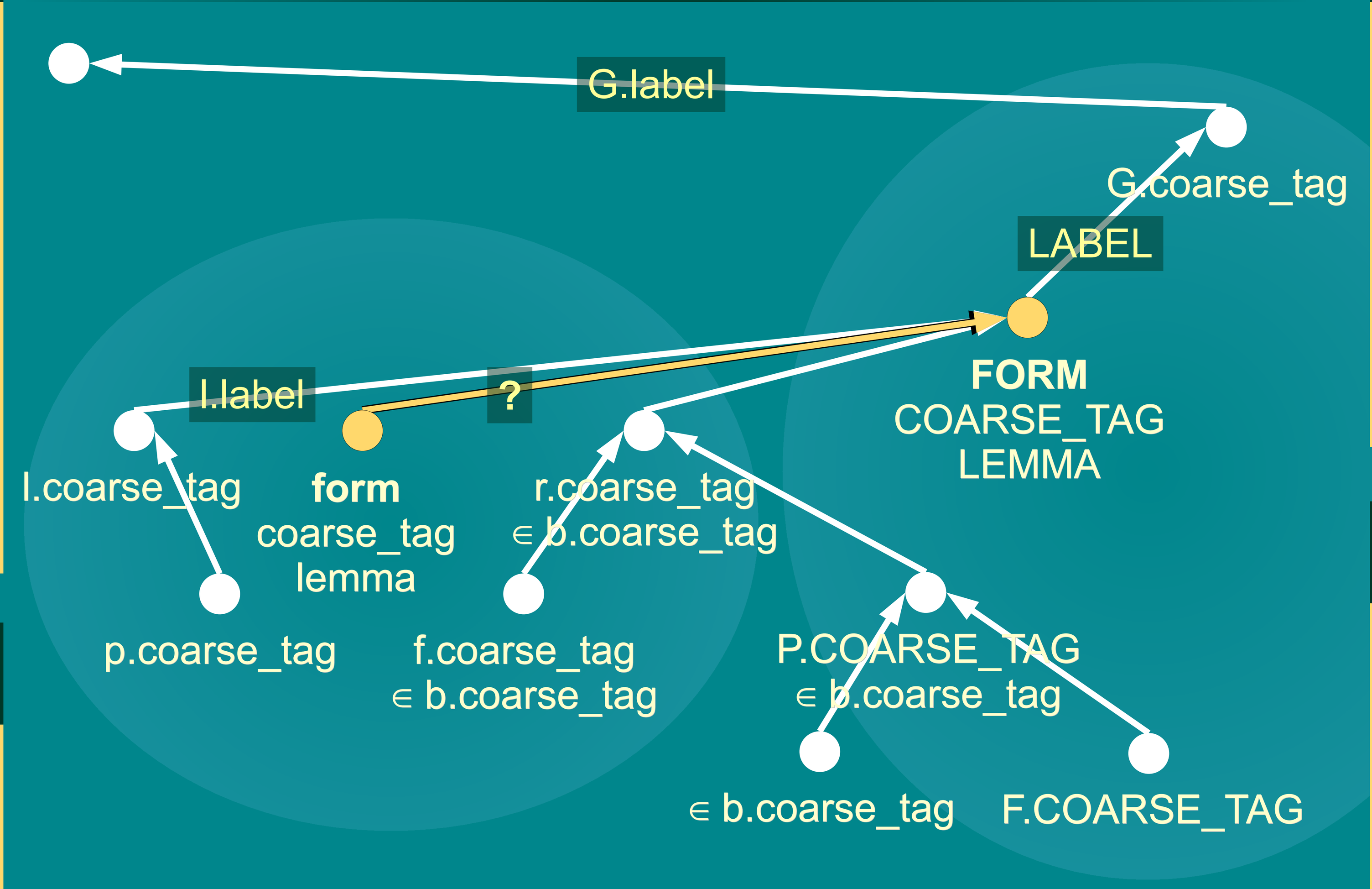
- perceptron-based online learning algorithm
- large-margin multiclass classification
- ultraconservative update: updates the weights so that the scores of the incorrect labels are lower than the score of the correct label at least by a given margin

Results

- 4 best systems for Czech from CoNLL 2009 Shared Task on labelled dependency parsing
- Labelled Attachment Scores (LAS) of the original systems
- LAS of the outputs of the systems with labels stripped and relabelled by our labeller

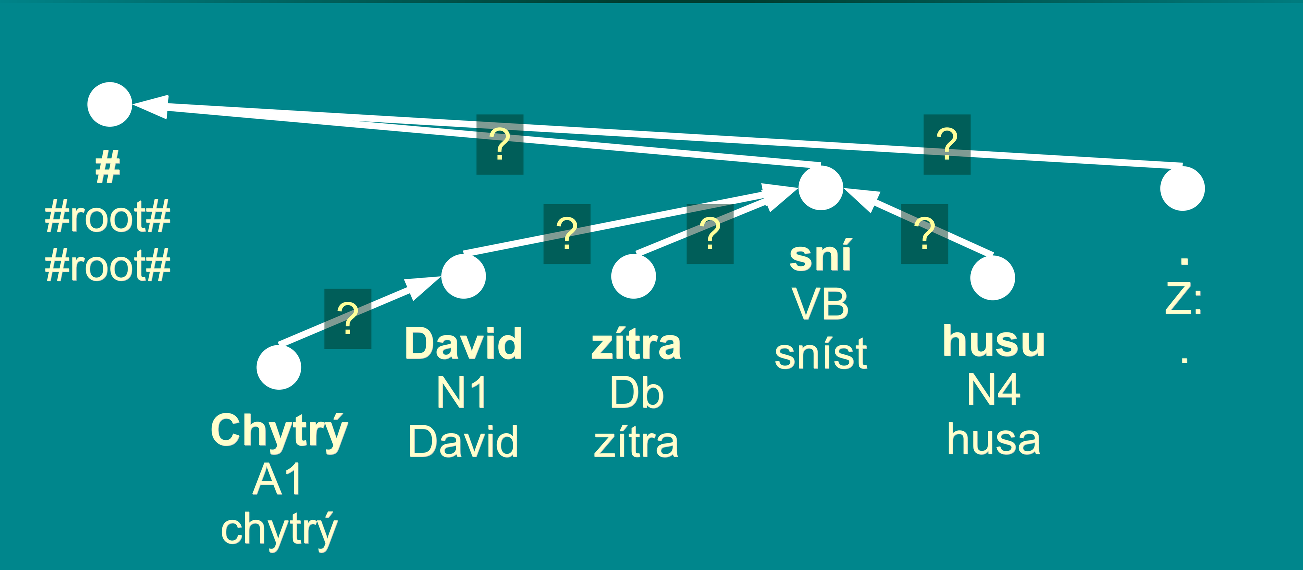
System	LAS original	LAS relabelled
merlo	**80.4%**	79.9%
bohnet	80.1%	81.0%
che	80.0%	**81.2%**
chen	79.7%	79.5%

Feature Functions



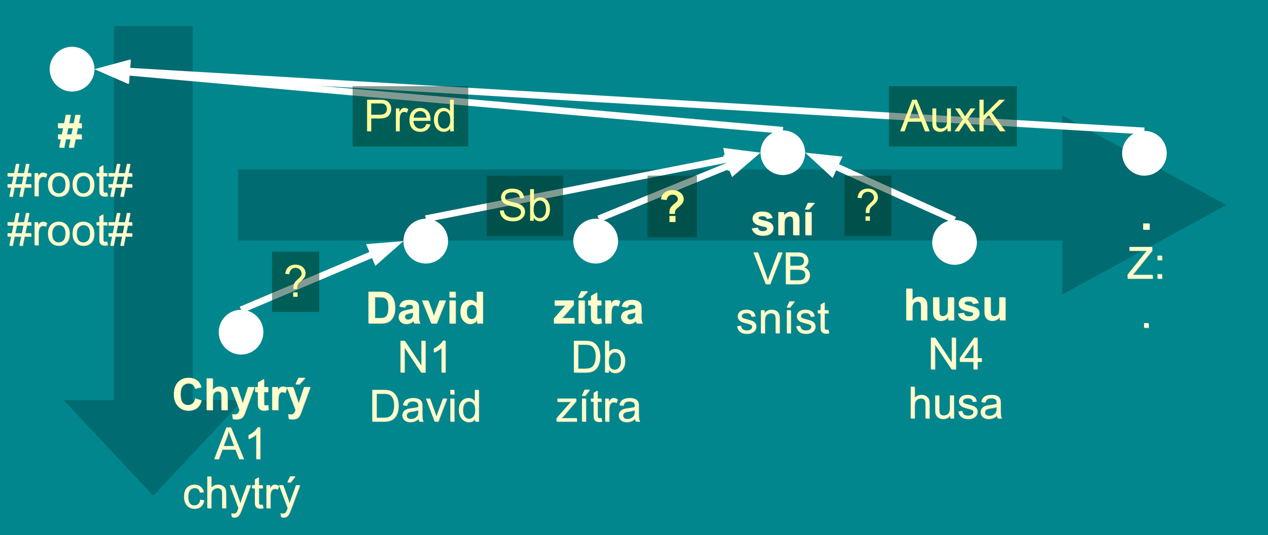
Unlabelled dependency tree

(Clever David tomorrow will-eat a-goose.)

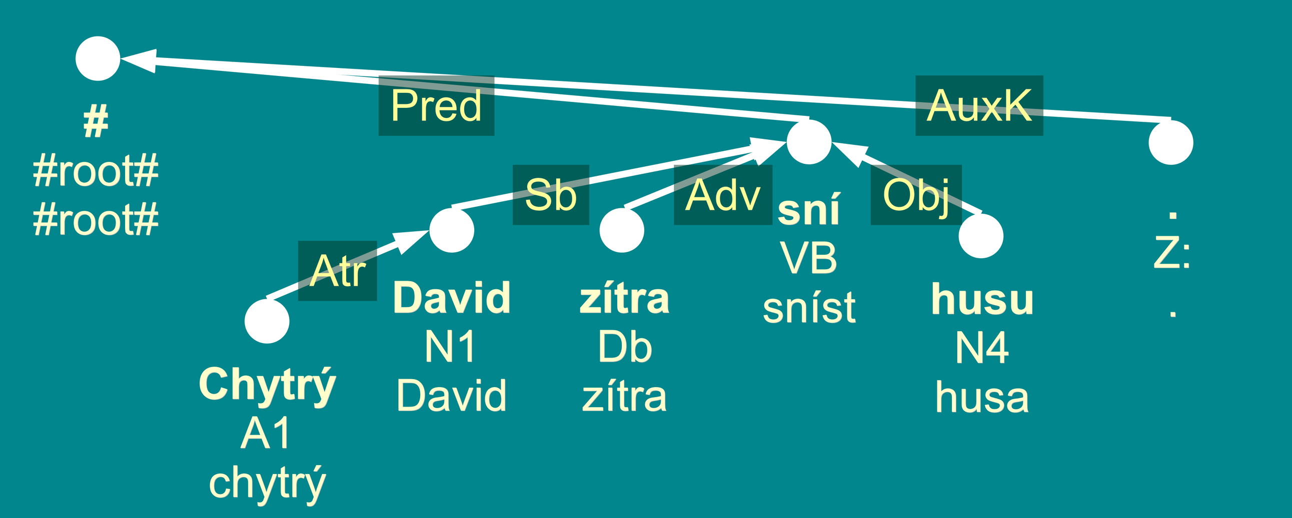


Labelling the edges

(top to bottom, left to right)



Labelled dependency tree



Feature Set

- 21 feature functions conjoined into 68 templates
- based on (McDonald 2005), (McDonald 2006) and (Carreras 2007)
- lowercase = child, UPPERCASE = parent
- p = preceding, f = following, b = between,
- l = left sibling, r = right sibling, G = grandparent

First-order Feature Functions

- local to the child – parent edge, not depending on the structure of the whole tree
- all (except for b.coarse_tag) computed both for parent and child

form: the word form of the child/parent

coarse_tag: part of speech and morphological case, or detailed part of speech if case is not exhibited

lemma: morphological lemma

p.coarse_tag: coarse_tag of the word immediately preceding the parent/child node

f.coarse_tag: coarse_tag of the word immediately following the parent/child node

b.coarse_tag: coarse_tag of each of the words between the parent node and the child node; this function can return multiple values, creating several features from one feature template

Higher-order Feature Functions

- use sibling edges and the grandparent node, use already assigned labels

LABEL: label assigned to the edge between the parent and the grandparent of the child node

l.label: label assigned to the left sibling edge, i.e. the edge between the parent and the left sibling of the child node

r.coarse_tag: coarse tag of the right sibling of the child node

G.coarse_tag: coarse tag of the grandparent of the child node

G.label: label assigned to the edge between the grandparent and the great-grandparent of the child node

G.attdir: whether the grandparent node precedes or follows the child node in the sentence

Non-local Feature Functions

childno: number of child nodes of the node

isfirstchild: whether the child is the first child of the parent

islastchild: whether the child is the last child of the parent

This research has been supported by the EU Seventh Framework Programme under grant agreement n° 247762 (Faust)

Text, Speech and Dialogue 2012, Brno, Czech Republic