A Appendix

Model	DM (%)	PAS (%)	PSD (%)
IPS +ML	27 (1.6%)	15 (0.88%)	25 (1.4%)
IPS +ML +RL	25 (1.5%)	19 (1.1%)	13 (0.76%)

Table 6: The number of graphs and the percentages of graphs with circles when not using heuristics to avoid circles. Results on the development sets. We also note circles are mostly small and local. So they do not affect other arc structures.

A.1 DAG formalism

Our proposed parsing algorithm possibly introduces circles in the resulting graphs. However, they are few in our experiments. Table 6 shows the number of graphs with circles and their relative frequency in our predictions for the development sets. Here we propose additional decoding rules to strictly prevent making these circles. Our decoding is iterative. Therefore when the newly arcs A are added to the existing partial SDP graph $\mathbf{y}^{\tau}$, the sum of $\mathbf{y}^{\tau} \cup A$ must not contain some circles. In that case, we swap arcs in A with other arcs that do not lead to circles, in the following way:

1 Search for arcs that make circles C in $\mathbf{y}^{\tau} \cup A$. If no circles (most in the case), return $\mathbf{y}^{\tau} \cup A$ as a partial SDP graph and go to the next transition.

2 Prepare buffers for reduced arcs $B(=\emptyset)$ and for alternative arcs $B'(=\emptyset)$.

3 From arcs in $C \cap A \backslash B$, choose one arc a that have the lowest probability of the softmax output and add to the buffer B.

4 For the reduced arc a, choose another arc (including `NULL`) that have the next largest probability of the softmax output of the arc a and add it to the buffer B'.

5 Check if $\mathbf{y}^{\tau} \cup A \backslash B$ have circles. If no circles, add B' to A and go to **1**. Otherwise, go to **3**.

The resulting graphs do not contain circles and form DAGs. This operation slightly improves the performance, but does not affect the results much, because circles are rare.

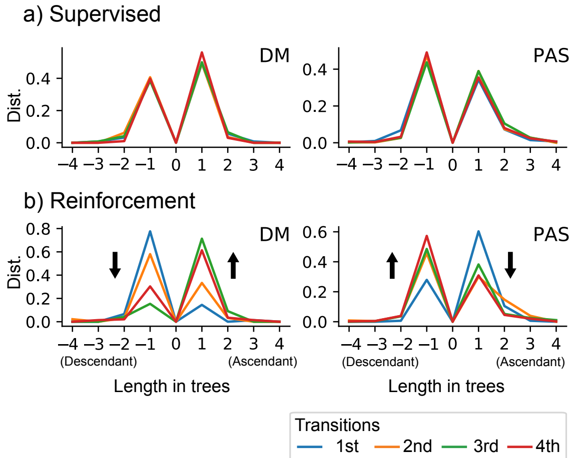

Figure 6: Arc length distributions on the WSJ syntactic dependency trees. (a) The length distribution of arcs with supervised learning (*IPS+ML*). (b) The length distribution of arcs with reinforcement learning (*IPS+ML+RL*). The four lines correspond to the first four transitions in the derivations. The horizontal axis corresponds to the length of the created arcs. The rightward of the horizontal axis corresponds to arcs from ascendants, while the leftward corresponds to arcs from descendants. The black arrows in the bottom figures illustrate the changes of distributions from the 1st to later distributions. They are in the opposite directions between DM and PAS.

A.2 Arc Length Analysis on WSJ Syntactic Trees

As a supplementary experiment, we wanted to explore the relation between the learned easy-first strategies and syntactic dependency lengths. The intuition was that attachments of length n that are consistent with syntactic dependencies may be easier than attachments of length n that are *not* consistent with syntactic dependencies, regardless of n. We therefore provide a quantitative analysis of the SDP parsing order using syntactic dependency trees of WSJ corpus as a reference. The syntactic dependency trees are extracted from WSJ constituency trees with the LTH Penn Converter.[8][9] For this, we consider the subset of undirected SDP arcs that match syntactic dependencies, i.e., where the semantic predicate words and the semantic argument words are in ascendant or descendant relations in the syntactic dependency trees. The directed SDP arcs that agree with the syntactic dependencies in direction, i.e., such that the syntac-

[8] http://nlp.cs.lth.se/software/treebank_converter/

[9] We note that we use these syntactic trees only for this supplementary analysis; and *not* for training or validating semantic parsers.

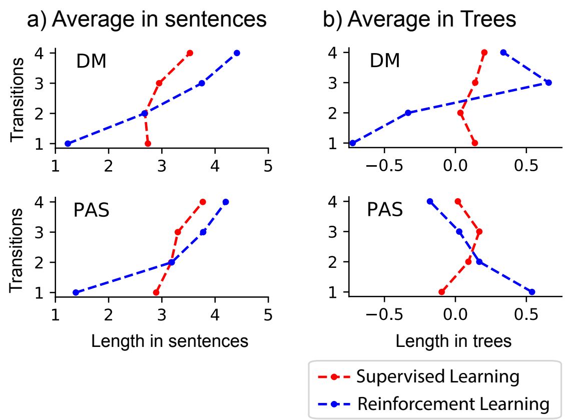

Figure 7: The average arc length comparisons of the supervised learning model (*IPS+ML*) and the reinforcement learning model (*IPS+ML+RL*). (a) Average arc length in sentences. The horizontal axis is the length of arcs in terms of words as of Fig. 4. The vertical axis corresponds to the first four transitions of models. (b) Average arc length in dependency trees. The horizontal axis is the same with Fig. 6.

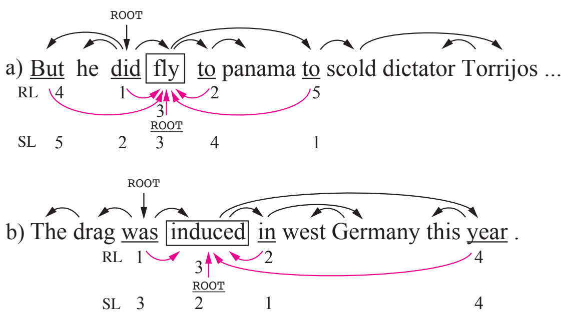

Figure 8: Examples of clauses parsed with PAS formalism. We present the gold syntactic trees above the sentences while the PAS arcs to a word under the sentence. We also attach the numbers of parsing orders. RL represents the parsing orders of the reinforcement learning model (*IPS+ML+RL*) while SL represents those of supervised leaning model(*IPS+ML*) on the PAS graphs.

tic head is also the semantic predicate, are said to have positive answer, as in the analysis above, while SDP arcs that are opposite have negative distance, because they go against the syntactic order. We show the distributions of length of arcs to semantic headwords that are created from the 1st to the 4th transitions from semantic argument words. We consider words that have four or more semantic headwords; this is because the models finish transitions for words that have fewer semantic headwords in the early transitions.

Fig. 6 presents the distributions of arc length from the 1st to the 4th transitions. These graphs are similar to Fig. 4, but only represent a subset of arcs, and now with a distinction between positive and negative arcs, depending on whether they agree with the syntactic analysis. We present the DM and PAS graphs, because we had too few examples for PSD. The graph (a) of supervised learning shows the same tendency from 1st to 4th transitions. The graph (b) of reinforcement learning shows a different picture: For PAS, the *IPS+ML+RL* model tends to resolve the short arcs that are consistent with the syntactic ones first (the blue line is higher in the positive span). In DM, however, the model tends to resolve the short arcs that are *inconsistent* with syntactic dependencies first.

Fig. 7 presents the averaged arc lengths at each transition step. The left column (a) is the average length of the created arcs at transition steps one to four. The right column (b) is the same relative to syntactic trees, using the same subsample as above and encoding disagreeing arcs with negative length values. In supervised learning, the averaged arc length does not vary much across transition steps, neither with respect to sentence positions nor with respect to syntactic trees. However, in reinforcement learning, the averaged arc length varies a lot across transition steps, relative to both sentence positions and agreement with syntactic headedness. The graphs in the (a) column suggest that the reinforcement learning model has a strong tendency to resolve adjacent arcs first. In the (b) column, we note that for reinforcement learning, the trajectories for DM and PAS are opposite. For PAS, early arcs are in line with syntactic dependencies, whereas for DM the opposite picture emerges.

Fig. 8 presents two example clauses with gold syntactic trees the partial SDP graphs, decorated with the parsing orders of our supervised baseline model (SL) and our reinforcement learning model (RL). RL resolves the adjacent words first, and the parsing orders of clauses (a) and (b) are consistent. The two PAS graphs and syntactic trees are relatively similar, but the arc directions are different, and the arcs from `ROOT` go to different words. The RL model prefers to create arcs that agree with the syntactic arcs first. We also note that the parsing orders of the SL models seem inconsistent across (a) and (b).