[lightly scribed by Luis and Siew Yeng]

Overview

DPF: Let me start with a high level overview. There are substantial resources on the wiki that should help with these topics.

One key feature from the box of delphin tools is the ability to help decide on the best parse/semantic representation for a tree. Looking at hundreds of thousands of examples, it can select the most likely place to attach a PP. eg. “i observed a cat from the house” - “cat from the house” or “observed the cat from the house”

This was done originally using a tool built by Rob Malouf, and it used to happen inside the LKB. And more recently Woodley created the Full Forest Treebanker, which has the capability of packing each and every parse of a very large forest of trees. This becomes rather critical when grammars become bigger (and produce a lot of ambiguity). This Maxent model is an essential component of using one of these medium to large grammars in DELPH-IN. It would be good if we could provide at least partially informed selections but DELPH-IN does not have the capability yet.

Maybe there are better parsing selection models out there (e.g. deep learning). But these have not been tested with great detail in DELPH-IN yet. However we do have good technology supporting the treebank portion of the pipeline.

There are other sets of tools that help deal with the other end of the spectrum. When we parse running text eg. newspaper text (etc.), we usually need to worry about tasks like sentence segmentation (there are tools out there to help with this). We (delph-in) have spent more time working on the word level.

There 3 levels that happen before the parsing happens (for English):

- Tokenization of the input sequence of characters that constitute a single sentence, using the space marker as a 1st guess as token delimitation. And then we do the same thing using punctuation with the enumeration of punctuation marks;
- Then we have a regular expression preprocessing step that tries to do a more specific job at this token segmentation. This is REPP, and should be in the wiki as ReppTop (https://github.com/delph-in/docs/wiki/ReppTop). This is a collection of linguistically identified rules designed by the grammarian to help the grammar. E.g. date expressions, currency values (e.g. AU$2.00). And this allows these to be treated in a more normalized fashion. We hope that it can be enumerated exhaustively.
- Finally we have chart mapping rules. E.g. ERG has a set of rules that tries to identify a set of rules for dates (6/6, or June 6). We take a token and decorate it with properties. Day of the month is a token that we would like to treat differently than others.
  E.g. “before June 16 people registered” -- this is a problem, we don’t want to force a single reading (don’t want to determine it in the preprocessing phase). Chart mapping allows to ambiguitate this. We can add some rules (e.g. if the number is from 1 to 31, then ambiguate it as a possible date).
  All these are known as Chart mapping rules (https://github.com/delph-in/docs/wiki/ChartMapping)
  This chart mapping is what is delivered to ACE, PET, etc, to start the parsing process.

DPF: Maybe Emily can say something about whether or not any of these are used in the Matrix.

EB: To my knowledge not much of this is used in the Matrix. But this is a good question: whether we can add some of this in the early Matrix output, so grammarians can be introduced to this earlier.

DPF: The problem of tokenization is more central to some languages (Chinese, Japanese) and they must be incorporated earlier in the process.

Glenn: Thai too..

DPF: Yes, I believe Thai doesn't even bother to tell you about sentence segmentation.

I wanted to lay out this broad ecosystem of what one can do to enable a linguistically interesting grammar to deal with raw data.

A couple other notes: The REPP is supported by all the platforms that we talked about, ACE, LKB, Agree, PyDelphin. This core basic mechanism has immediate support in all available platforms. Sadly this is not true for Chart Mapping, this capability is supported in ACE, PET and Agree, but it's not present in LKB (although this might change in the very near future, John and Stephan have wanted this in LKB for a long time now). The parse ranking is supported in many of these tools, but not in the LKB as well. On the other hand the ranking for realization is supported for the LKB.

Let's maybe open the floor to some questions. **Q & A:**

LMC: I think the area that is least well documented, and that I had most trouble with is the part where we use Chart Mapping to create Generic Entries. This was also a source of problems while tagging with the FFTB for ZHONG. FFTB has a feature/bug that is that sometimes you get a few trees remaining but you cannot fulfill them. That happens when there are more than one lexical entry under a lexical type. FFTB will not be able to disambiguate at the lexical level. ZHONG had many generic entries for dates etc. and was placing them all under the same lexical type. I tried to play with this (briefly) unsuccessfully. What I ended up doing was reduce the number of generic entries being used for dates (end up with 1 entry for dates regardless of format). Where is the best place to find documentation about building generic entries?

DPF: As far as I know there is no useful documentation for the generic entries quite yet. In the ERG, the ability to trigger the use of a Generic entry can be one in one of 2 ways:

- Look at the pos tag (e.g. Noun, Det, etc). For some words (e.g. "the") I have the lexical entries in the grammar, but for other (generic verb, generic transitive verb, singular noun, plural noun, etc) I am very happy to use the built-in tagger that ACE supplies to fill up the lexicon. E.g. "The glump walked into the room" the tagger tells me that "glump" should be a common noun. "Glump" is not an English word, but it should be treated as a common noun here.

- The second mechanism is to use the chart mapping rules to assign information to certain patterns. The chart mapping can put properties (called Classes) when creating chart entires. These are organized in a mini hierarchy, by different decorations/properties. The gle.tdl file (generic lexical entries) defines this hierarchy. Some people have successfully worked back from this file (e.g. Berthold), and disentangled what is packed there. But I should probably work on producing some documentation for this based on the English.

A different point that you raised is the fact that the FFTB is interested in the type but not the name of the lexical entry, and this can lead to ambiguities. For example, the downstream engines do not preserve the capitalisation (Ana and ANA can't be distinguished). And this was frustrating for me as a grammarian. They were both listed as proper nouns, and I wasn't sure (for a while) what was happening.

The LKB provides a nice tool that used to get rid of this in a debug tool, that will alert you about this (repeated entries under the same type). But for generic entries this is a real problem. One needs to find a way to tighten the chart mapping rules.

LMC: Just by defining new subtypes inside the proper grammar, I was not able to do that. Complete subtypes being inherited from the same type were being rejected.

DFP: We can work together to make better documentation.

Woodley: Using the new subtypes should have worked.

DPF: It would work if you were in the right hierarchy of types. But you might be in the wrong type space and it is confusing with my naming system. Send a couple of examples of subtypes you thought would have worked and it shouldn't be too hard to debug those and use those for the rest.

LMC: On this topic, if we known that having multiple senses (eg. 'society' with or without 'the', provided in a paper from Dan/Woodley) can lead to different expectations of whether it should have a determiner or not -- which is could be useful for mal-rules, I was wondering if there is any technical reasons within the tools for why this disambiguation at the lexical level is not supported.

Woodley: I believe there is not a technical problem. It was an intentional design, because that's what the *ancient regime* thought it should happen (with the belief that lexical entries are purely arbitrary and should never be looked at). But I agree that this could be used for looking at sense disambiguation. I think it's implemented but not really available. The branch where this was implemented was breaking unexpectedly. I should go back and look at that.

DPF: I am pretty sure we don't want to disambiguate everything at parsing time. So I think there needs to be some extension of the FFTB or a different tool to decorate the parse with different senses. This would be a deeper and interesting design challenge. Simply adding this capability to FFTB is not the solution for word sense disambiguation. One other challenge with this is that no one has come up with a naming convention that could be adopted by everyone. There have been a few attempts at predictable naming strategies.

LMC: I was not suggesting that all word sense disambiguation be available at the parse level. For the sense keys, I was starting to use the docstring, where I kept the different senses where an entry could have. Sometimes it could be of a different type but sometimes the same type.

DPF: I still want to discourage this WSD in the lexical entries. The power of combinatorics, you'll build a much larger forest -- millions of analyses instead of thousands of analyses. We should find a different way to include the results into the end result pipeline. I don't think we should be doing that, it is more work for the fftbanker.

TA: I also don't think the word sense should be brought into the lexicon. Luis do you have concrete examples of when you think it should?

LMC: If it is somehow useful for the parsing selection. Eg. one sense where PP attaches high and another sense where PP attaches low, then it might be worth it to do it to some scale. We should not only think about the (limiting) computing power we have today. This was just an idea to be tried by curious people who are stubborn like me.

EB: I think Dan's theory is that if there's no hard contrast then this should not happen at the lexical level... but Luis is asking: what about soft contrasts?

DPF: I appreciate that point, I think you might be right. I think it's reasonable to say "could this tool enable us to explore this", for a subset of a subpart of the grammar.

Woodley: Also, on combinatorics, the number of parses might go up exponentially, but packing happens linearly, so the time or size you take up to enable this does not explode in the same way.

DPF: You're right, since we're now living in this packed world, this might be different. In the *ancien regime* this would be difficult. But I should be open to someone exploring it.

Woodley: I wouldn't say it is definitely the right way to go, but if somebody wants to explore it, I want to find out how it goes.

LMC: I also wanted Roots. In the FFTB you cannot see the roots when you finish a sentence. And since I was using roots as a robust/mal-rule condition, this became a problem for people who didn't know the grammar well -- which was also why I wanted to retag the whole treebank (i.e. our students could see the root when tagging, and this led to some mistakes).

DPF: I think this can be channeled as a feature request. The derivation tree is stored, so this could be useful.

Glenn: Maybe we could make this an option for the tool.

Woodley: [back to senses] even though it is an interesting idea, it may be a slippery slope with high costs down the road.

DPF: We can find out if there will be unexpected black holes in the grammar.

LMC: I need to qualify that I am working in a more constrained context (learner's context). And that I was happy to live in a partially sense disambiguated world (where the words that I need for mal-rules etc are disambiguated) and partially not -- reducing the necessity to split senses for each lexical entry. Some sense distinctions might not be useful to keep in the grammar… but some could probably be useful, especially in the educational context.

DPF: I was starting to think that this might also work for Alexandre's proposal to map wordnet senses systematically. It is also part of my ambition to put those senses in, somehow. I could even think about proliferating these semi automatically and see what happens.

Woodley: One part of the machinery that doesn't necessarily scale well is the maxent model. Having a million times more ambiguity to select from (as negative examples), will be painful. If it becomes that the 100 best analyses are all the same sense, we may need to use FFTB mode as negative examples, and this will require a lot of resources. We currently use a n-best list but If there isn't enough variety, the model may not be able to learn the selection properly.

LMC: Can I ask how much memory are you currently using to train the maxent model based on the Redwoods?

Woodley: I am not entirely sure but what I know is that 500GB of RAM was not enough to have a 500 best parses (as negative examples), but was enough to do 250 best parses.

DPF: We could go find bigger machines in Oslo for example.

LMC: Yes, then we should read Emily's stochastic parrots ... again. But if a small scale is shown to be useful, then we try to do it. If we don't need the sense distinctions, we don't have them. But if we think we'll need them, we should have them.

Dan: I think this might be worth considering, yes.

LMC: Tuan Anh has also worked on adding senses to MRS. I was also thinking of working on coarse senses (i.e. not split if a 'burn' is caused by fire or by chemicals etc). Senses on Wordnet may not be useful for every single application. I was thinking of clustering the senses in a way that is useful for the context of my applications (i.e., education). Coarse senses should also probably be a part of this work.

TA: I mapped the wordnet to the ERG, and it didn't go well. On a small scale, you may feel the temptation to do that. But I don't think it is the way to go if you look at the full scale grammar.

DPF: At a minimum we would have to do some simplifying/aggregating of senses; but this would be expensive (human hours) because this would be done by hand.

Glenn: I thought the idea was to take the ones that could only affect the syntax.

DPF: No, now we are thinking of actually doing the opposite of that. Eg., different "bank" (river/financial institution) can build the same set of parse trees (hard constraints are the same), but the soft constraints are different (i.e. we could select the top parse differently). If we could encode some of those as first class properties in the grammar this could potentially happen.

Woodley: I'll make a prediction: the more complicated the model the more data you'll need to train it. So you might end up with a worse parse-ranking model.

LMC: But Dan was saying that because they're all from the same type, they would all be similar. You might be minimally diluting the things like PP attachment based on the senses. But I don't see how it would have a big hit on parse selection...

DPF: [it's a resource limitation] If sense disambiguation introduces more ambiguity, in order to maintain the same level of syntactic disambiguation, we may have to find more RAM.

LMC: Ah, yes, a resource limitation is definitely inherent.

Woodley: We can do it at a tradeoff for speed. And we could optimize the training too...

Glenn: Yes, if we don't have it work off of virtual memory…

DPF: Yes, using virtual memory is quite catastrophic. But ok, I've ran over the time we had allocated for this meeting. If no one else has any other questions [No? Good] I think we can adjourn the meeting here. Thank everyone:

Everyone: Bye

[as requester of this discussion, Luis thanks everyone for their participation]