Test suites, [incr tsdb()]

Ling 567
April 4, 2017

Overview

- Questions from Lab 1
- Evaluation and computational linguistics
- Evaluation and precision grammars
- Test suites and precision grammars
- Our test suites
- Features of [incr tsdb()]
- Look at Lab 2 instructions

Ask more questions!

- This class is not designed so that you can complete the work on your own with the information provided.

- I'm *relying* on you to ask questions, and not spend lots of time stuck. The 10 minute rule is for real!

- For Canvas, I try to answer very quickly --- but that means I can sometimes miss things if you put two questions in the same thread...

Questions from Lab 1: Formalism & features

- What's <! !>? How is it different from < >?
- Why do rules have both ARGS and DTR?
- What's the difference between features, types, and values?
- Do super types affect the constituents of a sentence or are do they merely provide inheritance for specific rules?
- Can subtypes override information from supertypes?
- Is there significance to the order of constraints inside [ ]?

Questions from Lab 1: Semantics

- Is knowing that _cat_n_rel is a noun relation helpful to me?
- Why is PNG a semantic feature?
- Why do we treat semantic content as read-only? What's the gain from having HOOK?
- Why do we have both C-CONT and SYNSEM.LOCAL.CONT? Do these not have the same content?
- What is XARG for?
- How are LKEYS set up?
- How does INDEX point to ARG0 if there's more than one relation?

Questions from Lab 1: Conventions & analyses

- rules.tdl and lrules.tdl seem like boilerplate. Is this an artifact of how the grammar is generated? I can see that we can define constraints for a rule or for a type. When is it a good idea to have several rules of the same type, rather than defining a new type for each?

- Why do you let specifiers select for their heads, but not subjects or complements?

- Would it be helpful for me to familiarize myself again with the rules outlined in 566?

- What are the possible values of the features?

- How do lexical rules work?

Questions from Lab 1: The Grammar Matrix

- What are some best practices to keep in mind when starting out constructing one of these grammars? The grammar we've been working with for this assignment appears massively complex, and it's difficult at this point to visualize the process of creating something of that nature within one quarter

- Will the rules that I create for my language be derived from grammars that are already created or will I be creating them from scratch?

- Why are there types defined as just one other type, with no other constraints? ex:

    - tr-verb-lex := nom-acc-transitive-verb-lex.

Questions from Lab 1: Etc

- How would we debug what's wrong with *Dance I*?
- How do we debug when there's multiple unification failures?
- Is there a way to use the LKB to see all of the supertypes of a given type? It seems like you can see all of the types that inherit FROM a given type, but I couldn't figure out how to go the other way.
- Is there a glossary?
- Are there any other books/references for lkb besides the given textbook?
- What makes good tdl style?

Questions from Lab 1: Tips

- With forcing unification failures, I found it helpful to first parse a similar sentence that did parse and then determine which rules were used. That helped to figure out what rules should have been used.

Overview

- Questions from Lab 1
- Evaluation and computational linguistics
- Evaluation and precision grammars
- Test suites and precision grammars
- Our test suites
- Features of [incr tsdb()]
- Look at Lab 2 instructions

Evaluation and Computational Linguistics

- Why is evaluation so prominent in computational linguistics?
- Why is it not so prominent in other subfields of linguistics?
- What about CS?

Intrinsic v. extrinsic evaluation

- Intrinsic: How well does this system perform its own task, including generalizing to new data?

- Extrinsic: To what extent does this system contribute to the solution of some problem?

- Examples of intrinsic and extrinsic evaluation of parsers?

Test data

- Test suites
  - Hand constructed examples
  - Positive and negative examples
  - Controlled vocabulary
  - Controlled ambiguity
  - Careful grammatical coverage

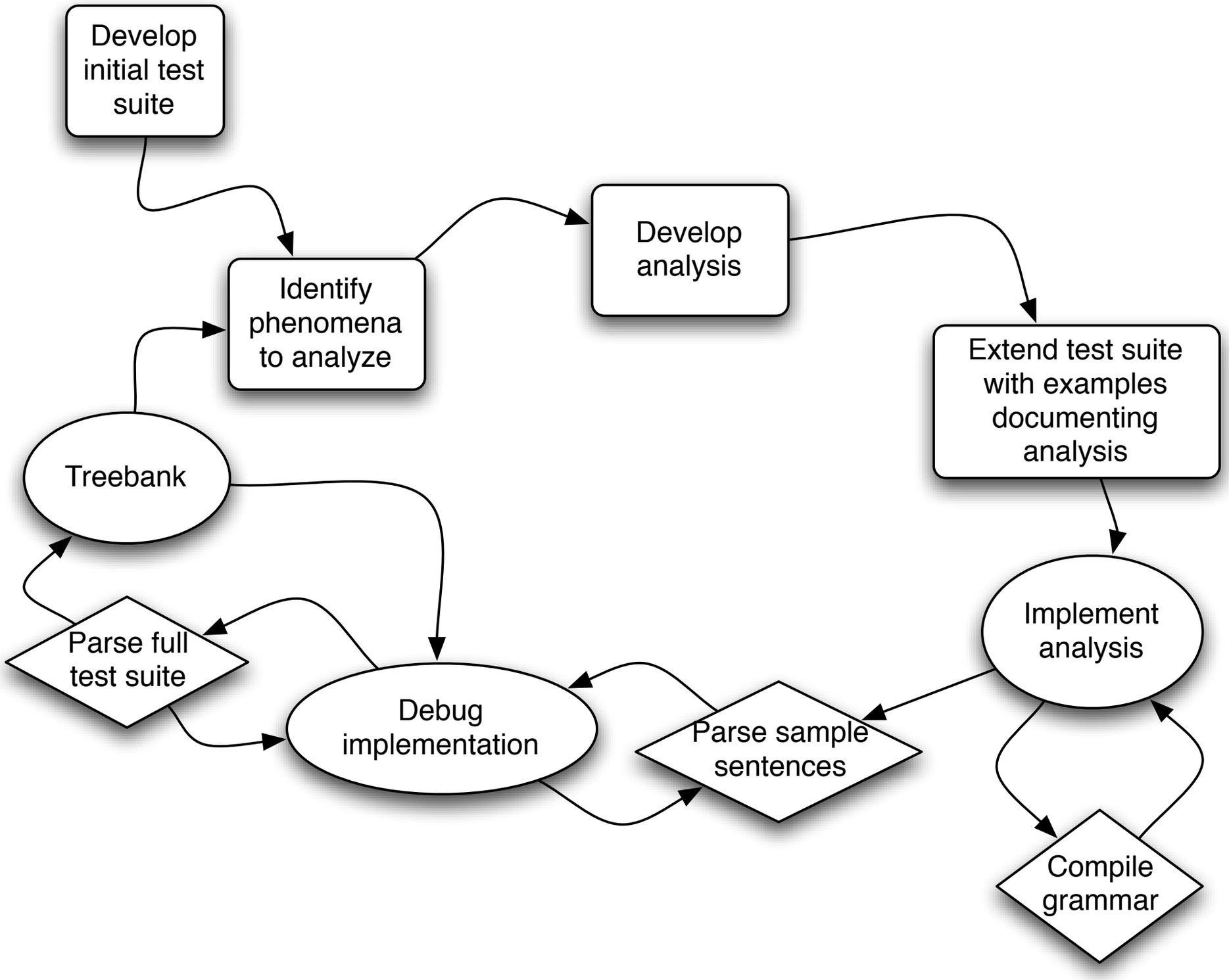
Test data

- Test corpora
  - Naturally occurring
  - More open vocabulary
  - Haphazard ungrammatical examples
  - Application-focused

Uses of test data

- How far do I have left to go?
  - Internal metric
  - Objective comparison of different systems
- Where have I been?
  - Regression testing
  - Documentation

Grammar engineering workflow



Evaluating precision grammars

- Coverage over some corpus
  - Which corpus?
  - Challenges of lexical acquisition
- Coverage of phenomena
  - How does one choose phenomena?
- Comparison across languages

Levels of adequacy

- grammaticality
- “right” structure
- “right” dependencies
- “right” full semantics
- only legit parses (how can you tell?)
- some set of parses including the preferred one
- preferred parse only/within first N

Our test suites

- Map out territory we hope to cover
- Include both positive and negative examples
- Serve as an exercise in understanding the description of the language
  - IGT format
  - Creating examples where necessary

On the importance of simple examples

- Why keep examples simple?
- How simple is too simple?
- What kinds of things make an example not simple enough?

On the importance of simple examples

- Awtuw [awt] (Feldman 1986:67)

(70) Yowmen Yawur du-k-puy-ey
Yomen Yawur DUR-IMPF-hit-IMPF
'Yowmen and Yawur are hitting (someone).' [awt]

- Basque [eus] (adapted from Joppen and Wunderlich 1995:129)

(112) Zuek lagun-ei opari polit-ak ema-ten dizkiezue.
you.PL.ERG friend-PL.DAT present nice-PL.ABS give-IMPF 3A.have.PLA.3PLD.2PLE
'You(pl) always give nice presents to your friends.' [eus]

On the importance of simple examples

- Russian [rus] (Bender 2013:92)

a. Человек укусил собаку.
Chelovek ukusi-l sobak-u.
man.NOM.SG.M bite-PAST.PFV.SG.M dog-ACC.SG.F
'The man bit the dog.' [rus]

[incr tsdb()] basics

- [incr tsdb()] stores test suite profiles as (plain text) relational databases: Each is a directory with a fixed set of files in it.

- Most files are empty.

- A profile that has not been processed has only two non-empty files: item (the items to be processed) and relations (always the same)

- Once the profile has been processed, the result of the processing is stored in some of the other files (in particular, parse and result)

[incr tsdb()] basics

- A test suite *skeleton* consists of just the item and relations files and can be used to create new test suite profiles

- [incr tsdb()] allows the user to compare two profiles to see how they differ

- It can also produce graphs plotting summary data from many profiles to visualize grammar evolution over time

- -> Demo

Overview

- Questions from Lab 1
- Evaluation and computational linguistics
- Evaluation and precision grammars
- Test suites and precision grammars
- Our test suites
- Features of [incr tsdb()]
- Look at Lab 2 instructions