Test suites, [incr tsdb()]

Ling 567
January 13, 2015

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 GoPost, 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: HPSG and analyses

- What's the deal with HOOK, C-CONT, XARG, LOCAL?
- What's a qeq?
- What's the difference between SPEC and SPR? (and SUBJ?)
- How does the RELS list relate to the chain of identities?
- Why put agree info on semantic indices (variable properties)?
- Where can I go to find out what all these features (and values) mean?
- What does cons stand for?
- Why so many more links in this chain than in 566?
- Will this grammar parse syntactically grammatical semantically bad sentences?

Questions from Lab 1: Formalism

- If # means that the values of two features referenced must be the same -- is it possible for that feature to be empty if it is a list? (Is it possible for it to be a list, or only for it to be an item in a list like < #item > ?) Could it be null or otherwise empty?

- In the case of feature-value pairs like "ASPECT aspect" and "MOOD mood", etc. (at least in the AVM view) are "aspect" and "mood" just sort of default values, meaning that there's no interesting aspect or mood in this sentence?

- What does the '.' mean in the tdl (for example, CONT.HOOK)?

Questions from Lab 1: Grammar Matrix

- What does it mean that the syntactic and semantic head daughters are different in the head-spec rule?
- In finding Keyrel's relation to Rel, what does norm-ltop-lex-item mean?
- How will we handle complex morphology?
- I'm hoping that when we start the analysis of our own language that we will be able to start at a VERY basic level and gradually build up so that I WILL in fact understand all of it!
- Will we be using the same matrix.tdl in our grammars?
- Why not just use regular lists?
- Do we really need all these types?

Questions from Lab 1: LKB software

- How can I learn more about the LKB software?
- What do all the files do? Why have irules.tdl/lrules.tdl in addition to my_language.tdl?
- Is there an IDE for HPSG/a better way to navigate grammars and feature structures?
- Does the parser work bottom-up?
- Are there more emacs short-cuts?

Questions from Lab 1: LKB Software

- When looking at one of my parse charts, "cats" had three different "N"s. The middle one said "plural-suffix", but the top one just said "cat". What is the purpose of the top "N"?

- What does the stuff that the LKB prints out in the emacs buffer when you parse a sentence mean?

- The subj-head-phrase structure rule inherits from basic-head-final rule which identifies the phrase's non-head and head daughters on the ARGS list.Oddly, I couldn't find the "ARGS" feature in the local avm of the node "S". Am I looking at the wrong place?

Questions from Lab 1: Grammar engineering

- This simple little grammar is huge --- how many of these types will we need to modify in Labs 5-9?

- This huge grammar can't even handle *dogs dance*, how can we make anything that will work?

- How many types will we implement from scratch?

- How will we develop our grammars? I know the questionnaire will handle most of that. But can we edit it manually in the .tdl files? It seems like edits are intuitive but could have unforeseen consequences because of the complexities of the grammar. Especially with the multiple inheritance hierarchy.

Questions from Lab 1: Other

- If something needs to be tweaked in the language info through the questionnaire, does the grammar have to be created again?

- What is the relationship between emacs and LKB? Is emacs a text editor or a linux terminal?

- Why does a 'sentence' not produce a unification error when dropped in as the first entry of subject head phrase rule's args list?

    - Example: A cat dances dog. Where 'a cat dances' is the sentence thrown in.

- Two trees were given when parsing "a cat chased me" (with one tree missing the 3rd-sg-noun edge). Which one should I choose?

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

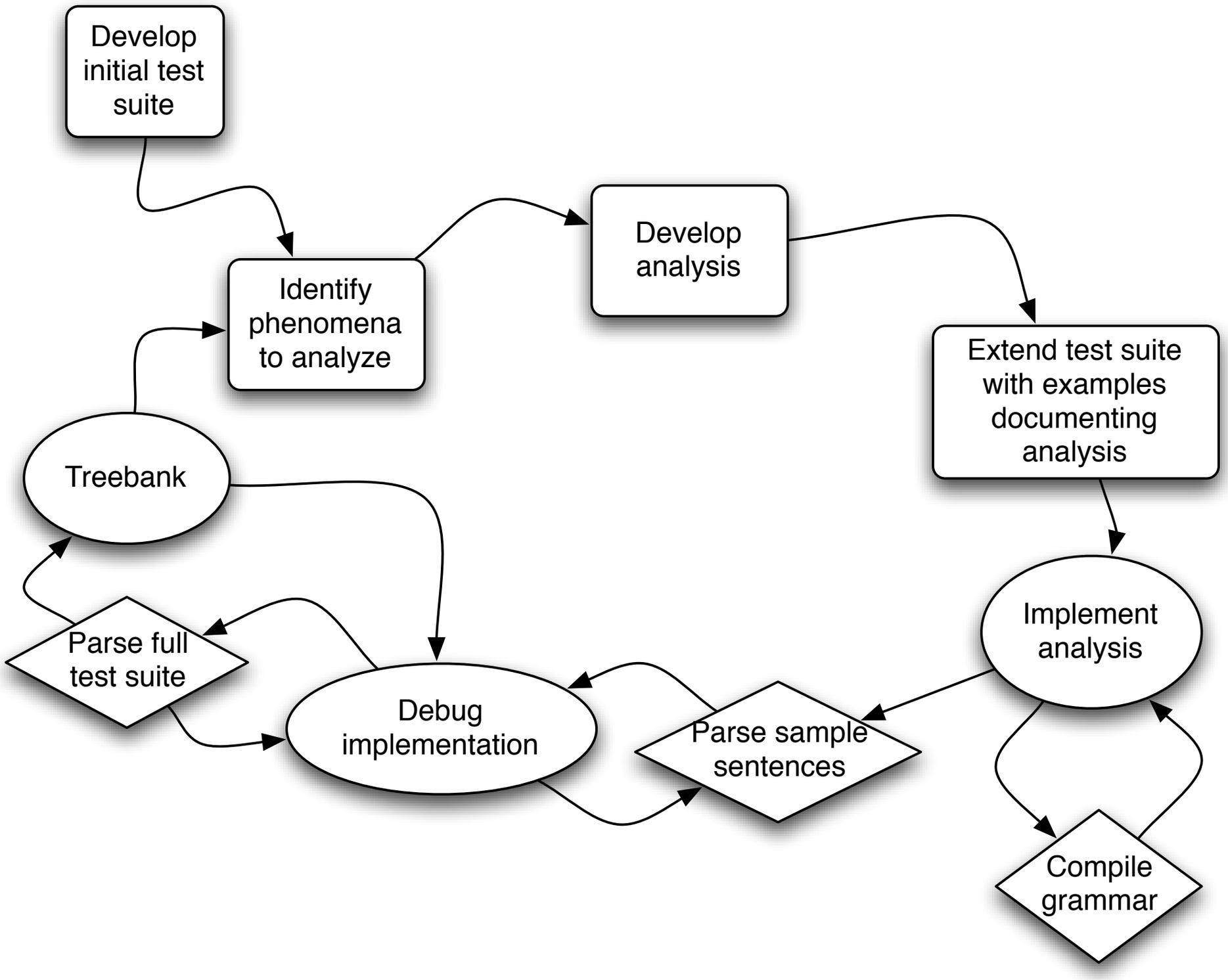
Test data

- Test corpora
    - Naturally occurring
    - More open vocabulary
    - Haphazard ungrammatical examples
    - Application-focused
- Which test data for which purposes?

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

[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