The Matrix: Future Directions
Wrap up

Ling 567
March 10, 2015

Overview

- Wrap up/reflections
- Matrix: Future directions
    - More libraries
    - More robust MMT
    - Applications, including language documentation

Goals: Of Grammar Engineering

- Build useful, usable resources
- Test linguistic hypotheses
- Represent grammaticality/minimize ambiguity
- Build modular systems: maintenance, reuse

Goals: Of this course

- Mastery of tfs formalism
- Hands-on experience with grammar engineering
- A different perspective on natural language syntax
- Practice building (and debugging!) extensible system
- Contribute to on-going research in multilingual grammar engineering

Reflections

- Where have the analyses provided by the Matrix (or suggested by the labs) seemed like a good fit?
- Where have they been awkward?
- What have you learned in this class about syntax?
- ... about knowledge engineering for NLP?
- ... about computational linguistics in general?
- ... about linguistics in general?
- What did working with a test corpus show you about the process of scaling to real-world text?

Feedback: Pair projects

- How did you divide the work?
- In what ways was having a partner helpful?
- Would you have learned more working on your own?

More reflections

- Semantic representations are important
    - It’s easier to work on them if they serve as an interface to something
- Analyses of phenomena interact
    - The more streamlined/motivated the analysis of each phenomenon is, the smoother the interactions
    - What interactions did you encounter?

More reflections: model and modeling domain

- From 566: Distinction between the model (HPSG grammar fragment) and the modeling domain (there: English).

- How did this play out in 567?

Future directions overview

- More libraries (and semantic harmonization)
- How this class will evolve
- MT: Auto-generated transfer rules, typological seeding of statistical NLP (including SMT)
- Lexical acquisition
- Ontological annotation
- AGGREGATION (and Ling 575)

More libraries

- New/updated this year: Adjectives (Trimble 2014)
- In progress: Valence-changing lexical rules
- Next up?
  - Demonstratives
  - Extensions/retrofits to questions, coordination
  - (more) extensions to word order
  - Other non-verbal predicates
  - Other intersective modifiers
  - Numeral classifiers
  - More verb subcategorization
  - Embedded clauses

How to make a library

1. Delineate a phenomenon
2. Survey the typological literature: How is this phenomenon expressed across the world's languages?
3. Review the syntactic literature for analyses of the phenomenon in its various guises
4. Design target semantic representations
5. Develop HPSG analyses for each variant and implement in tdl
6. Decide what information is required from the user to select the right analysis, and extend questionnaire accordingly
7. Extend customization script to add tdl based on questionnaire answers
8. Add regression tests documenting functionality
9. Add prose documenting how to use

How to evaluate a library

- Pseudo-languages
- Illustrative languages
- Held-out languages
- Test suites
- Choices files
- Error analysis

More libraries/reflection from current class

- What do you most wish was available in the customization system, based on what came up in your test suite?
- In your test corpus?

Evolution of 567

- New phenomena: ~~Wh-questions~~, relative clauses, while-clauses ...?
- Ever bigger jump start --- reaching the limit on this one?
    - Would working in groups of three make it possible to get to even bigger grammar fragments?
- Relatively new/how did these work out?:
    - Partnership with field linguists
    - Work with small corpora
- Coverage-driven labs seem most satisfying (MT demo, corpus coverage). Is this true? Can the course be rebalanced to do more of this?

Lexical acquisition

- How can we import lexical entries from other linguistic resources (e.g., FIELD lexicons, ODIN, other IGT collections)?

- How big do the grammars have to get before we can embark on (semi-)automated lexical acquisition?

- To what extent do the lexical properties of translational equivalents predict lexical properties in another language?

- How can we most effectively leverage human effort?

- How do we know when we're missing an appropriate type?

Autogenerated transfer rules

- Identify “grammaticized” differences in MRSs
- “Publish” choices along these dimensions for each grammar
- Create a library of transfer rules from property to property:
  - pro-drop to pronouns (and vice versa)
  - mismatches in demonstrative distinctions
  - can <> the possibility exists
  - hurt/cause feel+pain/cause harm

Autogenerated transfer rules

- Use language-specific pred values
- Create transfer rules on the basis of PanDictionary or other lexical resources
- Measure the extent of translation divergence (Francesca Gola's MS thesis)
- Use bitexts and statistical methods to detect word pairs requiring more than straight pred-mapping transfer rules

Ontological annotation

- Annotate grammars with links to GOLD (Farrar & Langendoen 2003)
    - Locate which constraints contribute to which phenomena
    - Index analyses for discovery in grammars and treebanks
- Annotations in Matrix core
- Annotations in customization system
- Support for user annotation

AGGREGATION: Research goals

- Precision implemented grammars are a kind of structured annotation over linguistic data (cf. Good 2004, Bender et al 2012).

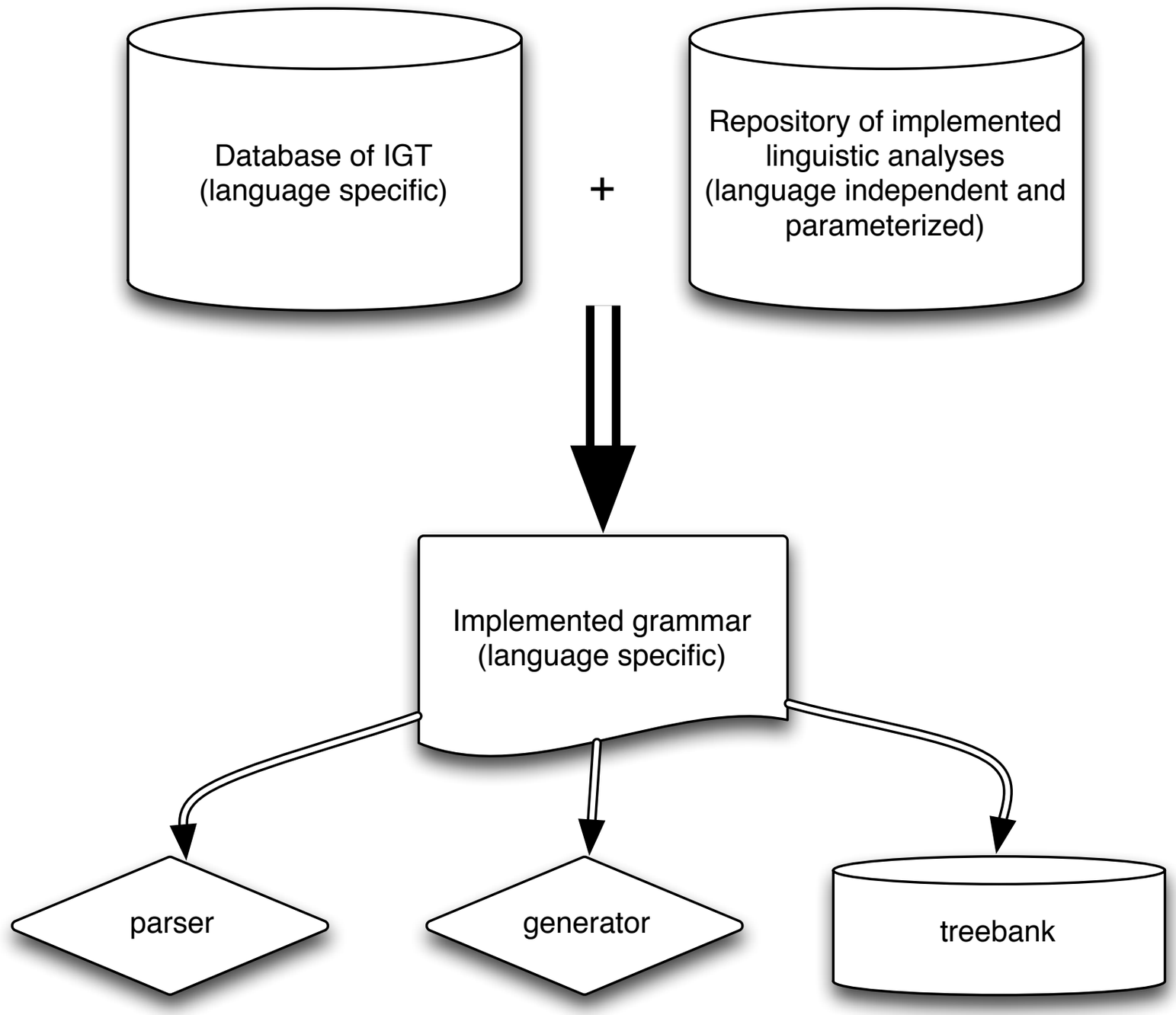
- They map surface strings to semantic representations and vice-versa.

- They can be used in the development of *grammar checkers* and *treebanks*, making them useful for language documentation and revitalization (Bender et al 2012)

- But they are expensive to build.

- The AGGREGATION project asks whether existing products of documentary linguistic research (IGT collections) can be used to boot-strap the development of precision implemented grammars.

Combining linguistic knowledge

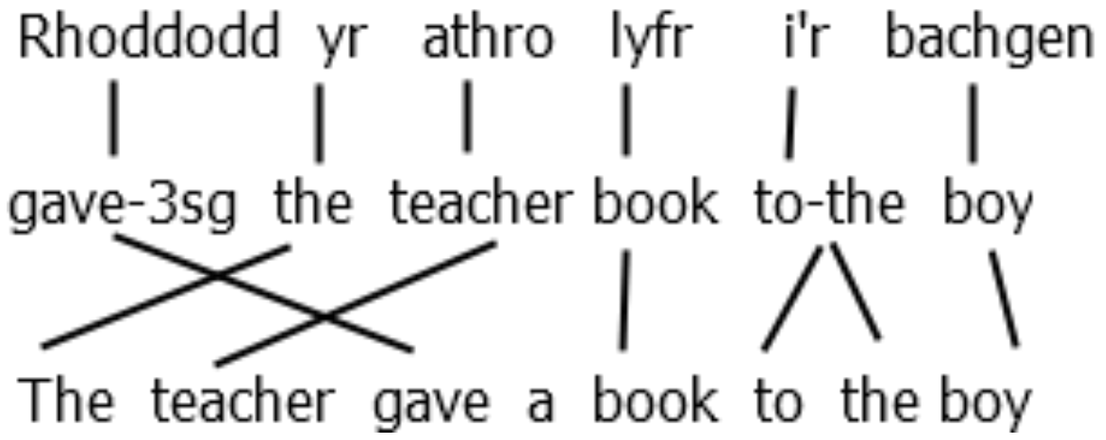


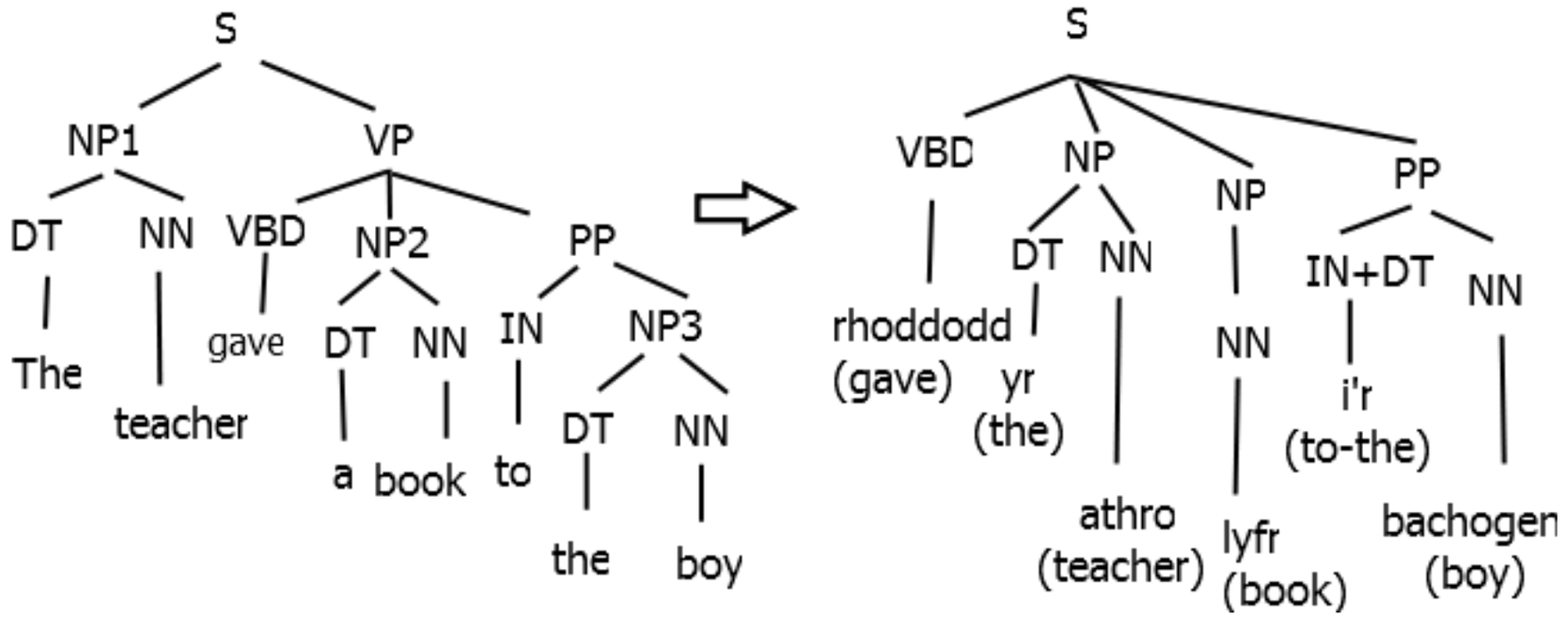
RiPLes: Goals

- RiPLes: information engineering and synthesis for Resource Poor Languages
- Support rapid development of NLP resources for RPLs by bootstrapping through IGT
- Support cross-linguistic study through creating ‘language profiles’ based on IGT analysis

(Xia & Lewis 2007, Lewis & Xia 2008)

RiPLes: IGT projection methodology





(Xia & Lewis 2009)

RiPLes: Results

Table 3: Experiment 1 Results (Accuracy)

	WOrder	VP +OBJ	DT +N	Dem +N	JJ +N	PRP$ +N	Poss +N	P +NP	N +num	N +case	V +TA	Def	Indef	Avg
basic CFG	0.8	0.5	0.8	0.8	1.0	0.8	0.6	0.9	0.7	0.8	0.8	1.0	0.9	0.800
sum(CFG)	0.8	0.5	0.8	0.8	0.9	0.7	0.6	0.8	0.6	0.8	0.7	1.0	0.9	0.762
CFG w/ func	0.9	0.6	0.8	0.9	1.0	0.8	0.7	0.9	0.7	0.8	0.8	1.0	0.9	0.831
both	0.9	0.6	0.8	0.8	0.9	0.7	0.5	0.8	0.6	0.8	0.7	1.0	0.9	0.769

Table 5: Word Order Accuracy for 97 languages

# of IGT instances	Average Accuracy
100+	100%
40-99	99%
10-39	79%
5-9	65%
3-4	44%
1-2	14%

(Lewis & Xia 2008)

Word order options

- Lewis & Xia 2008, Dryer 2011 (WALS)
  - SOV
  - SVO
  - OSV
  - OVS
  - VSO
  - VOS
  - no dominant order
- Grammar Matrix
  - SOV
  - SVO
  - OSV
  - OVS
  - VSO
  - VOS
  - Free (pragmatically determined)
  - V-final
  - V-initial
  - V2

Word order in the Grammar Matrix

- More than a simple descriptive statement

- Affects phrase structure rules output by the system, but also interacts with other libraries (e.g., argument optionality)

- These phrase structure rules help model the mapping of syntactic to semantic arguments

- Underlying word order is not reflected in every sentence; testsuites won't have the same distribution as naturally occuring corpora

- Matrix users advised to choose fixed word order if deviations from that order can be attributed to specific syntactic constructions

Methodology

- Parse English translation and project the parsed structure onto the language line (per RiPLes)

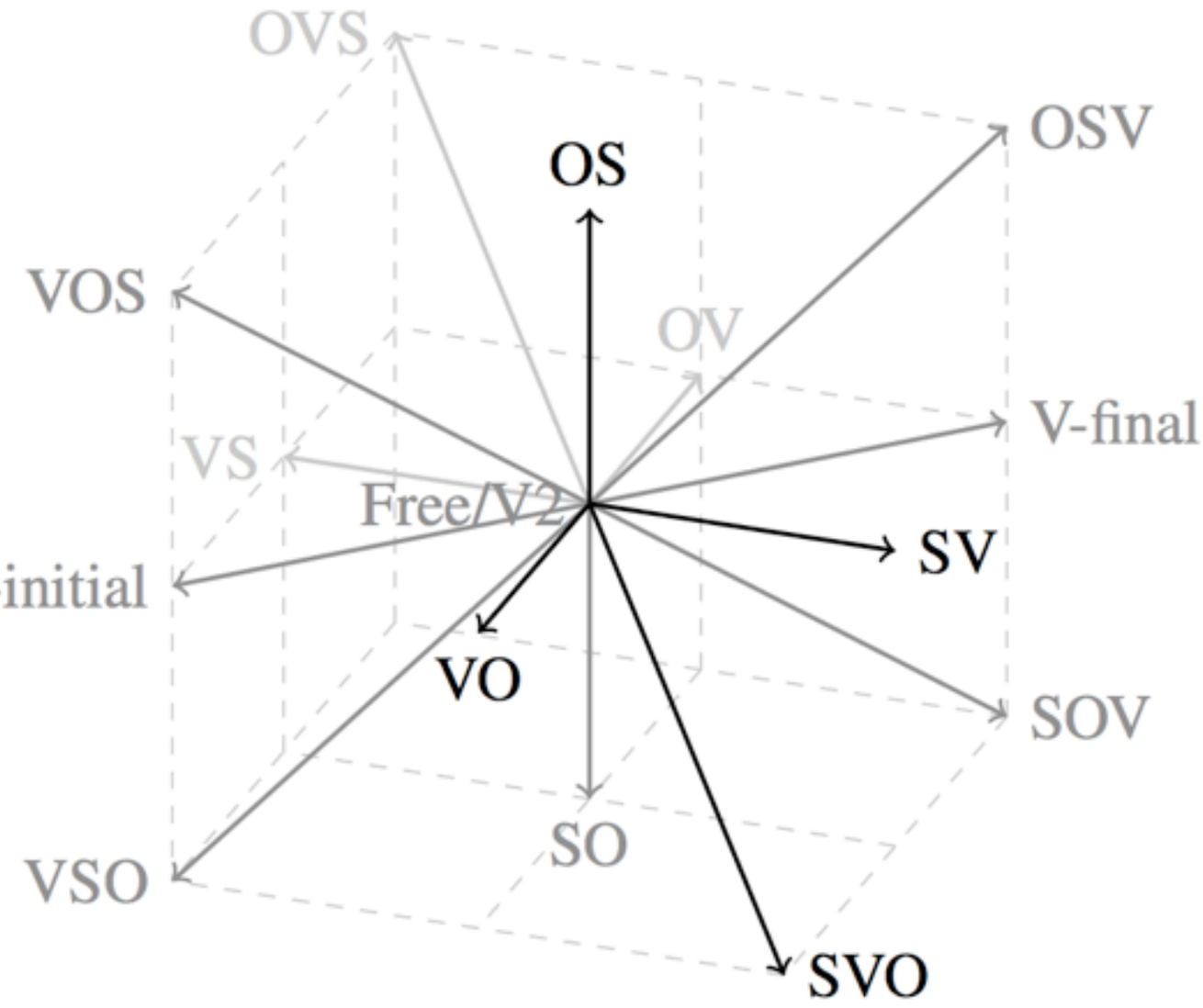
- Add -SBJ and -OBJ function tags to the English parse trees (by heuristic), and project these too

- *Observed word orders:* counts of the 10 patterns SOV, SVO, OSV, OVS, VSO, VOS, SV, VS, OV, and VO in the source language trees

- Decompose SOV, SVO, OSV, OVS, VSO, VOS into order of S/O, S/V and O/V

Methodology

- SOV, SVO, OSV, OVS, VSO, VOS
- Measure Euclidean distance to positions of canonical word orders
- In a separate step, distinguish free from V2



Dev and test data

- 31 testsuite + choices file pairs, developed in Linguistics 567 at UW (Bender 2007)

	DEV1	DEV2	TEST
Languages	10	10	11
Grammatical examples	16–359 (median: 91)	11–229 (median: 87)	48–216 (median: 76)
Language families	Indo-European (4), Niger-Congo (2), Afro-Asiatic, Japanese, Nadahup, Sino-Tibetan	Indo-European (3), Dravidian (2), Algic, Creole, Niger-Congo, Quechuan, Salishan	Indo-European (2), Afro-A Austro-Asiatic, Austronesi Arauan, Carib, Karvelian, N. Caucasian, Tai-Kadai, I

Results

- Compare to most-frequent-type (SOV, Dryer 2011)

Dataset	Inferred WO	Baseline
DEV1	0.900	0.200
DEV2	0.500	0.100
TEST	0.727	0.091

- Sources of error:
  - Testsuite bias
  - Misalignment in projections

Case system options in the Grammar Matrix: Case marking on core arguments of (in)transitives

- None
- Nominative-accusative
- Ergative-absolutive
- Tripartite
- Split-S
- Fluid-S
- Split conditioned on features of the arguments
- Split conditions on features of the V
- Focus-case (Austronesian-style)

- The choice among these options makes further features available on the lexicon page, including case frames
- There is always the option to define more cases and case frames

Two methods

- GRAM: Assume Leipzig Glossing Rules-compliance (Bickel et al 2008)
- Search gloss line for case grams, and assign system as follows:

Case sysem	Case grams present NOM ∨ ACC	ERG ∨ ABS
none		
nom-acc	✓	
erg-abs		✓
split-erg	✓	✓

(conditioned on V)

- SAO: Use RiPLes to identify S, A, and O arguments
- Collect most frequent gram for each
- Compare most frequent grams across S/A/O to determine case system

Results

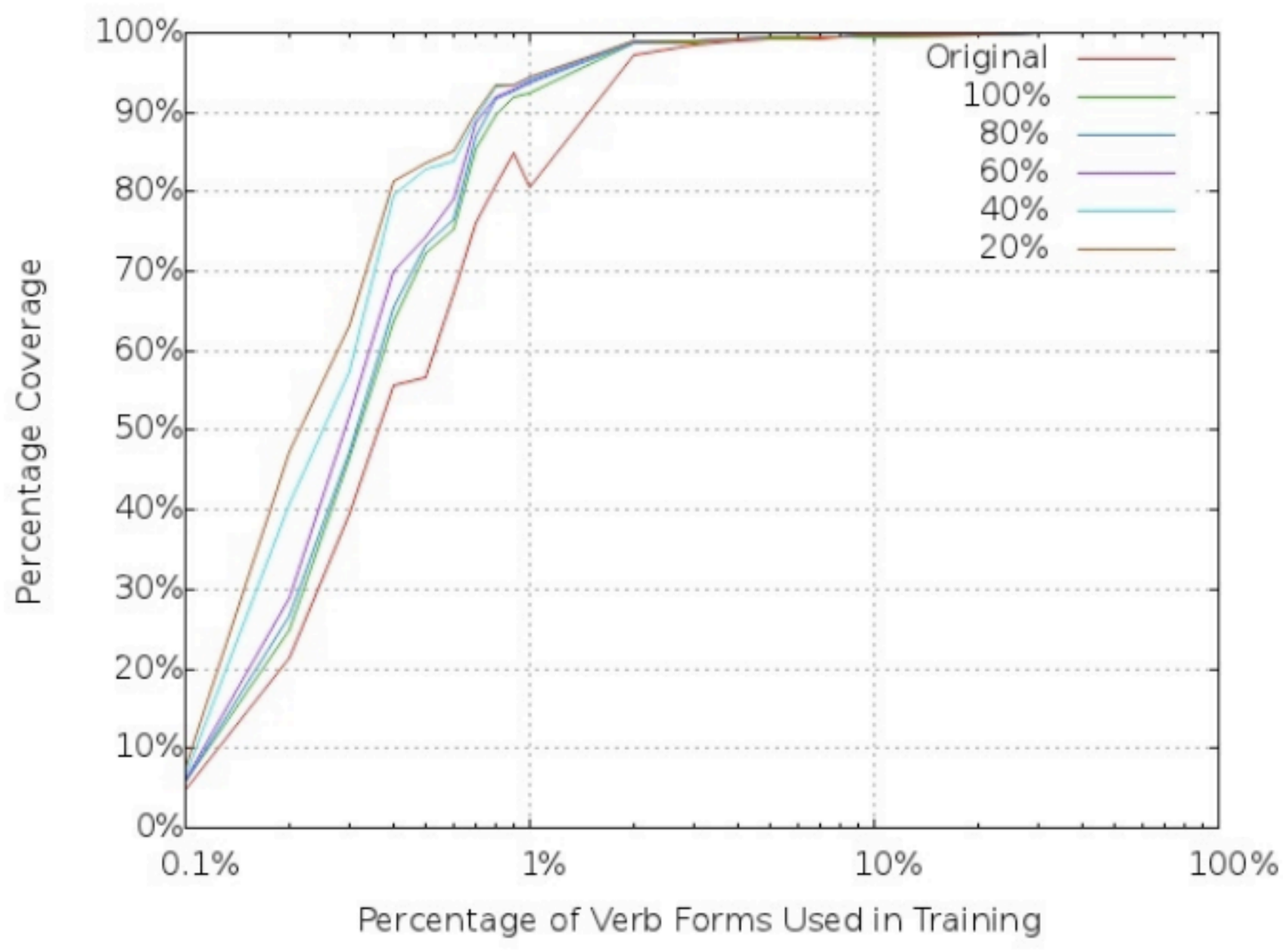
Dataset	GRAM	SAO	Baseline
DEV1	0.900	0.700	0.400
DEV2	0.900	0.500	0.500
TEST	0.545	0.545	0.455

- GRAM confused by non-NOM/ACC style glossing
- SAO confused by testsuite bias (spurious most-frequent elements)
- SAO confused by alignment errors (e.g. case marking adpositions)

MOM: Matrix-ODIN Morphology

- David Wax MS thesis (near completion)
- Use RiPLeS-like methodology to identify verbs
- Use GIZA++ again to align morphemes to glosses
- Extract lexical rule definitions: input, form, features (in some cases)
- Compress lexical rules into shared position classes based on shared inputs
- Output choices files

MOM: Results (French)



MOM Results: ODIN data (Turkish, Tagalog)

% input overlap	No. Choices	Verb Classes	Position Classes	Coverage
Baseline	N/A	N/A	N/A	0.803
No Compression	1445	**43**	62	0.794
100%	1356	43	22	**0.824**
80%	1356	43	22	0.824
60%	1356	43	22	0.824
40%	1346	43	17	0.824
20%	**1342**	43	**15**	0.824

Table 6.8: Coverage of Test Data for Tagalog

% input overlap	No. Choices	Verb Classes	Position Classes	Coverage
Baseline	N/A	N/A	N/A	0.459
No Compression	3975	**137**	168	0.630
100%	3716	137	52	0.668
80%	3696	137	42	0.668
60%	3696	137	42	0.668
40%	**3639**	137	**18**	**0.674**
20%	3639	137	18	0.674

Table 6.9: Coverage of Test Data for Turkish

Bender et al 2014: End-to-end, for Chintang [ctn]

Choices file	# verb entries	# noun entries	# det entries	# verb affixes	# noun affixes
ORACLE	900	4751	0	160	24
BASELINE	3005	1719	240	0	0
FF-AUTO-NONE	3005	1719	240	0	0
FF-DEFAULT-GRAM	739	1724	240	0	0
FF-AUTO-GRAM	739	1724	240	0	0
MOM-DEFAULT-NONE	1177	1719	240	262	0
MOM-AUTO-NONE	1177	1719	240	262	0

Table 2: Amount of lexical information in each choices file

	Training Data (N = 8863)				Test Data (N = 930)			
choices file	lexical coverage (%)	items parsed (%)	items correct (%)	average readings	lexical coverage (%)	items parsed (%)	items correct (%)	average readings
ORACLE	1165 (13)	174 (3.5)	132 (1.5)	2.17	116 (12.5)	20 (2.2)	10 (1.1)	1.35
BASELINE	1276 (14)	398 (7.9)	216 (2.4)	8.30	41 (4.4)	15 (1.6)	8 (0.9)	28.87
FF-AUTO-NONE	1276 (14)	354 (4.0)	196 (2.2)	7.12	41 (4.4)	13 (1.4)	7 (0.8)	13.92
FF-DEFAULT-GRAM	911 (10)	126 (1.4)	84 (0.9)	4.08	18 (1.9)	4 (0.4)	2 (0.2)	5.00
FF-AUTO-GRAM	911 (10)	120 (1.4)	82 (0.9)	3.84	18 (1.9)	4 (0.4)	2 (0.2)	5.00
MOM-DEFAULT-NONE	1102 (12)	814 (9.2)	52 (0.6)	6.04	39 (4.2)	16 (1.7)	3 (0.3)	10.81
MOM-AUTO-NONE	1102 (12)	753 (8.5)	49 (0.6)	4.20	39 (4.2)	10 (1.1)	3 (0.3)	9.20

Table 3: Results

Summary

- First steps towards our long-term goal: Automatically create working grammar fragments from IGT, by taking advantage of
    - Grammar Matrix customization system's mapping of relatively simple language description files to working grammars
    - Linguistic analysis encoded in IGT
    - RiPLes methodology for further enriching IGT
- Resulting grammars are of interest for testing the Grammar Matrix as a set of typological hypotheses
- And potentially for field grammarians (when built-out) as they can support the creation of treebanks and exploration of corpora for unanalyzed phenomena

Ling 575 Spring 2015

- Current AGGREGATION inference scripts do not take full advantage of the Grammar Matrix customization system

- Term projects: Adding inference for various parts of the customization system

- Test data: Large datasets for Chintang [ctn] and Matsigenka [mcb]

- Test data: 567 testsuites + choices files (see Language CoLLAGE)

Overview

- Wrap up/reflections
- Matrix: Future directions
    - More libraries
    - More robust MMT
    - Applications, including language documentation
- Next time: MMT extravaganza and course evals