User-Oriented Solving and Explaining of Natural Language Logic Grid Puzzles

Jens Claes¹, Bart Bogaerts², Rocsildes Canoy², Tias Guns²

¹jensclaes33@gmail.com
²Vrije Universiteit Brussel, firstname.lastname@vub.be

https://bartbog.github.io/zebra/
Holy Grail-ish

From *human-level* problem specification, to *human-level* solving and explanation.

applied on

Logic Grid Puzzles\(^1\)

How?

- Clues in Natural Language
- Logic Grid
- Visual explanation of reasoning steps

\(^1\)https://freuder.wordpress.com/pthg-19-the-third-workshop-on-progress-towards-the-holy-grail/
Zebra Tutor, a Holy Grail

“Our design choices:

- Input: natural language (with semi-automated processing)
- Reasoning: Blackburn & Bos semantic parsing + IDP solver
- Output: visual explanation
  - Abstractions: grid visualisation and clues
  - Ordering of reasoning steps: by 'mental effort',
  - in practice: order by number of clues used, then by number of facts used

“From human-level problem specification to human-level solving and explanation.”
HolyZebra approach

Clues → Pos Tagging → Chunking & Lexicon building → First order Logic → IDP Language → Explanation Generation → Visualisation

JSON .json → .py → .py → .pl → .idp → .json
POS tagging

In: Natural Language sentences
Out: Part-Of-Speech tagged words

Technically: NLTK's Perceptron tagger with the Penn Treebank POS set

"The patient who was prescribed Enalapril is not Heather"¹

¹Output: (the, DT), (patient, NN), (who, WP), (was, VBD), (prescribed, VBN), (enalapril, NN), (is, VBZ), (not, RB), (heather, NNPN).
In: POS tagged sentences

( the, DT ), ( patient, NN ), ( who, WP ), ( was, VBD ), ( prescribed, VBN ), ( enalapril, NN ), ( is, VBZ ), ( not, RB ), ( heather, NNPN )

Mid: Chunking

( the, det ), ( patient, noun ), ( who, relpro ), (( was, prescribed ), tvGap ), ( enalapril, pn ), (( is, not ), cop ), ( heather, pn )

Out: Lexicon for our B&B grammar

... noun([patient], [patients]),
    pn([heather]),
    pn([enalapril]),
    tvGap([was, prescribed], [for, their, heart, condition], [prescribe]),
...
Chunking and lexicon building

In: POS tagged sentences

Mid: Chunking

Out: Lexicon for our B&B grammar (next slide)

Old school NLP approach:
· regular expressions
· semi-automated

Difficulty:
· custom vocabulary per puzzle
· word-play by authors
Blackburn and Bos framework as a base:

- Defined grammar based on 10 other puzzles, which includes:
  - Template sentences specific to logic grid puzzles
  - alldifferent rules: "Of X, Y and Z, one is..."
  - Numerical comparisons ("John scored 3 points higher than Mary"), ...

- Extended Blackbrun & Bos framework to reason about types:
  - Each entity (John, points) has a type
  - Some relations (scored, has more, received) are synonyms: types allow detecting them
To IDP language

Input: Logical Representation (Discourse Representation Theory)
Output: IDP Puzzle specification

1. Compute interpretation of different types
   - Type deduction from grid (if available)
   - Type inference from sentence(s).

“The Englishman *smokes* cigarettes”

“The person who owns a dog does not *smoke* cigars”
**To IDP language**

**Input:** Logical Representation (Discourse Representation Theory)

**Output:** IDP Puzzle specification

1. Compute interpretation of different types
   - Type deduction from grid (if available)
   - Type inference from sentence(s).
     ❗ also supports missing entities (e.g. the zebra)
To IDP language

Input: Logical Representation (Discourse Representation Theory)
Output: IDP Puzzle specification

1. Compute interpretation of different types

2. Build Vocabulary
   - Types and relation for each transitive verb or preposition
   - Ensure at least 1 relation between each 2 types
     ✅ Important for explanation

   “John lives in the red house” ➔ LivesIn(<person>, <house>)
To IDP language

Input: Logical Representation (Discourse Representation Theory)
Output: IDP Puzzle specification

1. Compute interpretation of different types

2. Build Vocabulary

3. Construct IDP Theories:
   1. Translate each clue into IDP language
   2. Add implicit constraints present in logic grid puzzles:
      - Synonymy
      - Bijection (lives_in / owns_house)
      - Transitivity (rel1(A,B) rel2(B,C) → rel3(A,C))
To IDP language

Input: Logical Representation (Discourse Representation Theory)
Output: IDP Puzzle specification

1. Compute interpretation of different types
2. Build Vocabulary
3. Construct IDP Theories:
   4. Solving the Puzzle using the IDP solver
      (ASP / model expansion / lazy clause generation-like solver)
To explanations

Ordering of reasoning steps by mental effort required

get_reasoning_step(S: current partial assignment):

Until a solve leads to propagation (a more strict partial assignment):

Try: solve S + all implicit constraints

For n=0..|clues|, for all subsets of clues of size n:

Try: solve S + the constraints from the subset of clues

Break if it lead to propagation

For each literal that was assigned during propagation:

Compute minimal partial assignment S' needed to derive the literal

→ the S' is the UNSAT core when negating the literal

Store (S', clues used, literal)

return (S', clues, literal) with smallest S'
Visualisation

Of tatum and annabelle, one earns 144000 per year and the other lives in the cyan house

1 https://bartbog.github.io/zebra/
Related work

Solving Logic Puzzles: From Robust Processing to Precise semantics
Iddo Lev¹, Bill MacCartney¹, Christopher D. Maning², and Roger Levy²,
Workshop on Text Meaning and Interpretation, January 2004
¹Department of Computer Science, Stanford University {iddolev|wcmac|manning}@cs.stanford.edu
²Department of Linguistics, Stanford University rog@stanford.edu

Similarities
Solver: FOL reasoner
Semantic Logic language
Compositional semantics: Blackburn&Bos

Differences
Data: GRE¹ and LSAT² multiple-choice logic puzzles with 1 correct answer:
Backtracking ambiguities: ranking of possible output representations
Statistical parser
Generic semantic rules applicable to other problem settings
Use of Theorem prover and model builder to solve problem (parallel execution, first to solve the problem).

¹Graduate Record Exam, ²Law School Admission Test
Related work

LogicSolver – Solving Logic Grid Puzzles with POS Tagging and First-Order Logic.
Ross Nordstram, Masters Project, December 2016, University of Colorado rnordstr@uccs.edu

Key Differences

Goal: POS Tagging, First-order Logic
Puzzler as a base reasoner system

Normalization
- NER (named entity recognition)
- Structure detection

Parsing
- Link Grammar Parser
- Regex (clues) pattern matching

Main Problems

Hard-coded clue structure knowledge to identify comparisons (regex).

Solver

Hard-coding of less vs greater comparisons and comparison contexts

Ex1:
- Taller must apply to entity type of “height” or “distance”

Ex2:
- Jeffery’s pack is larger than the Grennel pack
Conclusion

From *human-level* problem specification to *human-level* solving explanation.

Our design choices:

- input: natural language, semi-automated processing
- reasoning: Blackburn & Bos semantic parsing + IDP solver
- output: **visual** explanation
- abstractions: grid visu and clues
- ordering: by mental effort, proxy = nr of literals used

*Can also serve as 'help' function when user is stuck*
Conclusion and future work

From *human-level* problem specification to *human-level* solving explanation.

Our design choices:
- input: natural language, semi-automated processing
- reasoning: Blackburn & Bos semantic parsing + IDP solver
- output: **visual** explanation
- abstractions: grid visu and clues
- ordering: by mental effort, proxy = nr of literals used

Can also serve as 'help' function when user is stuck

- Better NLP: statistical techniques?
- Explanation orderings and proxies for 'mental effort'
- Explanation abstractions, e.g. important parts of clue
- Other puzzle explanations
- Applicability in industrial problems, e.g. scheduling?