

University of Groningen

PDRT-SANDBOX

Venhuizen, Noortje; Brouwer, Harm

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version

Publisher's PDF, also known as Version of record

Publication date:

2014

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Venhuizen, N., & Brouwer, H. (2014). *PDRT-SANDBOX: Implementing Projective Discourse Representation Theory*. Poster session presented at 18th Workshop on the Semantics and Pragmatics of Dialogue (SemDial-DialWatt 2014), Edinburgh, United Kingdom.

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Projective Discourse Representation Theory (Venhuizen et al., 2013; in prep)

An extension of DRT (Kamp, 1981; Kamp & Reyle, 1993) with **projection variables** to account for projection phenomena, e.g., **presuppositions** (Venhuizen et al., 2013) and Potts' (2005) **conventional implicatures** (Venhuizen et al., 2014).

Projection variables indicate the interpretation site of semantic content, so all content stays *in situ* during composition.
> Projected content is indicated by free pointers, or pointers bound to the label of a non-local, accessible PDRS.

Challenge: projection variables affect the **construction procedure** and definition of **accessibility** non-trivially.

input/output formats

PDRT-SANDBOX: a Haskell library that implements PDRT and DRT

A modular and flexible NLP library that incorporates machinery to build, combine, and translate structures from PDRT as well as DRT ... and **much more!**

```

Prelude Data.PDRS> let johnisvegan = PDRS 1 [(1,2)] [PRef 2 (PDRSRef "x")] [PCon 2 (Rel (PDRSRef "John") (PDRSRef "x"), PCon 1 (Rel (PDRSRef "vegan") (PDRSRef "x")))]
Prelude Data.PDRS> let notjohnlikesmeat = stringToPDRS "<math>\neg \lambda x. \lambda y. \{2, John(x)\}, \{3, meat(y)\}, \{3, like(x,y)\}, \{3, 2\}>"
Prelude Data.PDRS> Debug notjohnlikesmeat
PDRS 1 [] [] [PCon 1 (Neg (PDRS 3 [(3,2)] [PRef 2 (PDRSRef "x"), PRef 3 (PDRSRef "y")] [PCon 2 (Rel (PDRSRef "John") (PDRSRef "x"), PCon 3 (Rel (PDRSRef "meat") (PDRSRef "y"), PCon 3 (Rel (PDRSRef "like") (PDRSRef "x", PDRSRef "y"))])))]
Prelude Data.PDRS> Set johnisvegan
-1, {-2, x}, {-2, John(x)}, {-1, vegan(x)}, {(1,2)}
Prelude Data.PDRS> Boxes notjohnlikesmeat
[[-1]]
  [3]
  2 - x 3 - y
  1 -
    2 - John(x)
    3 - meat(y)
    3 - like(x,y)
    3 = 2
Prelude Data.PDRS> Linear johnisvegan
1: [-2, x], {-2, John(x)}, {-1, vegan(x)} [(1,2)]
Prelude Data.PDRS> !t johnisvegan
johnisvegan :: PDRS
Prelude Data.PDRS> johnisvegan == notjohnlikesmeat
False
  
```

(sandbox syntax)

(set-theory)

(boxes)

(linear)

data types

```

Prelude Data.PDRS> printAMerge johnisvegan notjohnlikesmeat
[[-1]] + [3] = [4]
2 - x      2 - John(x)      2 - x
2 - John(x)  2 - x 3 - y      2 - John(x)
1 - vegan(x) 3 - meat(y)      4 - vegan(x)
1 = 2      3 = 2            4 = 2

Prelude Data.PDRS> printAMerge (PDRS 1 [] [PRef 1 (PDRSRef "y")] [PCon 1 (Rel (PDRSRef "Mary") (PDRSRef "y"))]) (PDRS 3 [] [] [PCon 3 (Rel (PDRSRef "vegetarian") (PDRSRef "y"))])
[[-1]] + [3] = [3]
1 - y      3 - vegetarian(y)  1 - y
1 - Mary(y) 3 = 1            1 - Mary(y)
3 = 1      3 = 1            3 = vegetarian(y)
3 = 1      3 = 1            3 = 1

Prelude Data.PDRS> let maryisvegetarian = PDRS 1 [] [PRef 1 (PDRSRef "y")] [PCon 1 (Rel (PDRSRef "Mary") (PDRSRef "y"))] <<<< PDRS 3 [] [] [PCon 3 (Rel (PDRSRef "vegetarian") (PDRSRef "y"))]
Prelude Data.PDRS> Linear $ maryisvegetarian <<<< johnisvegan
4: [-1, y], {-2, x}, {-1, Mary(y)}, {-4, vegetarian(y)}, {-2, John(x)}, {-4, vegan(x)} [(4,1), (4,2)]
Prelude Data.PDRS> !t maryisvegetarian <<<< johnisvegan
maryisvegetarian <<<< johnisvegan :: PDRS
  
```

merging

lambda-abstractions

```

Prelude Data.PDRS> let because pdr1 pdr2 = PDRS 5 [] [PRef 5 (PDRSRef "p1"), PRef 5 (PDRSRef "p2")] [PCon 5 (Prop (PDRSRef "p1") pdr1), PCon 5 (Prop (PDRSRef "p2") pdr2)], PCon 5 (Rel (PDRSRef "because") (PDRSRef "p1", PDRSRef "p2"))]
Prelude Data.PDRS> !t because
because :: PDRS -> PDRS -> PDRS
Prelude Data.PDRS> because
[5]
Ak8.kk1.
5 - p1 5 - p2
5 - p1: k8
5 - p2: k1
5 - because(p1,p2)
Prelude Data.PDRS> Linear (because johnisvegan notjohnlikesmeat)
5: [-5, p1], [-5, p2], [-5, p1]: [-2, x], [-2, John(x)], [-1, vegan(x)], [(1,2)], [-5, p2]: 1: [-1, -3], [-2, x], [-3, y], [-2, John(x)], [-3, meat(y)], [-3, like(x,y)], [(3,2)], [-5, because(p1,p2)]=]
Prelude Data.PDRS> pdrsToDRS notjohnlikesmeat (PDRT to DRT)
x
- y
- meat(y)
- like(x,y)
John(x)
(PDRT to FOL)
Prelude Data.PDRS> pdrsToFOL notjohnlikesmeat
3x[-3y(meat(w,y) A like(w,x,y)) A John(w,x)]
  
```

translations

(variable binding and renaming!)

Implementation

Accessibility of PDRT referents is defined using a **projection graph**.

Unresolved lambda-(P)DRSs (Musken, 1996) as Haskell functions.

Various **input** (set-theoretic, sandbox syntax, Boxer's Prolog syntax; Bos, 2003), and **output** formats (*pretty printing* of (P)DRS representations, and non-recursive P-Tables).

Application

Forming an NLP toolchain together with a syntactic parser (à la C&C tools and Boxer; Curran et al., 2007), or reasoning about semantic representations using a theorem prover or model checker.

