# 証明論的アプローチを用いた整合性判定による照応解析手法の提案

小斉平ひな　　高橋優太　　戸次大介
お茶の水女子大学
{g1820513, takahashi.yuta, bekki}@is.ocha.ac.jp

## 概要

近年の意味解析における照応解析では，語の間の意味的類似度を用いて照応表現 (apaphoric expression) の先行詞を同定するといった深層学習・機械学習によるアプローチが主流となっている．一方，依存型意味論 (Dependent Type Semantics, DTS) は，依存型理論による自然言語の意味論の一つであり，特に照応 (anaphora) と前提 (presupposition) は DTS が説明できる意味現象の中枢となっている．この論文では，DTS の未指定型 (underspecified types) を用いて照応表現の可能な先行詞を全て探索し，その前後の文脈から照応が可能な先行詞のうちどの対象が整合性が取れているかを判定する手法を提案する．また，提案する手法が，どれだけの照応解析の問題を解くことができるかを評価する．

## 1　はじめに

DTS では近年，未指定型 (underspecified types) と呼ばれる新たな理論的装置が導入された [1]．DTS における照応解析は，未指定型を介した証明探索として再定式化され，未指定型を介した照応解析が実装された [2]．

一般的に照応解析とは，英語を例にとれば，テキストの中にある *it* や *he* などの照応表現の先行詞を同定するタスクである．一方，DTS に限らず形式意味論における照応理論のタスクは（照応表現の先行詞を同定することではなく）どの表現が照応表現の先行詞になりうるか（到達可能性）を予測すること，および照応表現の到達可能なすべての表現について，その表現が先行詞である場合の意味を与えることである．

本研究の目的は，DTS の照応理論のそのような性質を利用して，照応表現の先行詞を同定する手法を提案・実装することである．到達可能な表現が照応表現の先行詞であるか否かは，そこから得られた意味表示に前後の文脈との整合性があるか否かで判定することができる．

証明支援系 (proof assistants) の一つである Coq [3] を用いて，DTS の未指定型を介した照応解析を実装する．まず，Coq が提供する `refine` タクティクと本研究が実装する自動照応探索タクティクを用いて到達可能な先行詞を全て探索する．次に，照応表現を含むテキストに照応表現に到達可能な先行詞を当てはめたテキストを生成する．最後に，接続詞と用法ごとに分類した手法により生成したテキストの推論を行い，推論が証明可能であるか証明不可能であるかにより，対象の前後の文脈との整合性を判定する．

## 2　照応探索手法

未指定型 $\begin{bmatrix} x@A \\ B \end{bmatrix}$ を介した照応探索手続きを，Coq を用いて次の手順で定式化する．

**(i)** 到達可能な先行詞を探索する．

**(ii)** 照応表現を含むテキストに到達可能な先行詞を当てはめたテキストを生成する．

まず，未指定型を Coq の帰納的型 `aspT` として定義する．

```
Inductive aspT (A : Type) (a : A) (B : Type) :
    Type :=
  resolve : B -> aspT A a B.
```

ここで型 `aspT` は，型 $A$, 型 $A$ をもつ項 $a$, 型 $B$ の三つから構成される複合的な型 `aspT` $A$ $a$ $B$ である．ただし，型 $B$ は項 $a$ に依存していてもよい．例として以下の談話を考える．

**(1)** The lawyer asked the witness a question, but he was reluctant to answer it.[1)]

**(1-0)** There is a lawyer, a witness, and a question.

1）この文は [4] の例文 10. である．

**(1a)** The lawyer asked the witness a question.

**(1b)** He was reluctant to answer it.

(1) の談話では，*he* と *it* が照応表現であり，aspT を用いて (1b) の意味表示を次のように表す．

```
aspT {x:entity & human x -> False} ?[asp1]
(aspT {y:entity & man x} ?[asp2]
(reluctant (answer (projT1 ?asp1))
           (projT1 ?asp2)))
```

?[asp] は空所であり aspT の型 $A$ を持つ項である．(1a) のテキストから (1-0) のように *a lawyer* と *a witness* と *a question* が前提され，それぞれは次のような型をもち，常識的な背景知識を公理として追加する．

*a lawyer* の意味表示：

$$\begin{bmatrix} x : \textbf{entity} \\ \textbf{lawyer}(x) \end{bmatrix}$$

*a witness* の意味表示：

$$\begin{bmatrix} x : \textbf{entity} \\ \textbf{witness}(x) \end{bmatrix}$$

*a question* の意味表示：

$$\begin{bmatrix} x : \textbf{entity} \\ \textbf{question}(x) \end{bmatrix}$$

公理として追加する常識的な背景知識：

```
Hypothesis LawyerHuman : forall x:entity,
  lawyer x -> human x.
Hypothesis WitnessHuman : forall x:entity,
  witness x -> human x.
Hypothesis QuestionNotHuman : forall x:entity,
  question x -> (human x -> False).
```

Coq が提供する refine タクティクを (1b) の意味表示に適用して，?[asp] 以外の部分に型検査を施しつつ，空所?[asp1] と?[asp2] を埋めることをゴールに設定する．照応解析は，空所を用いて表された照応表現をゴールに設定し，前提から照応の指示対象を探索するという証明探索と言い換えることができる．ここで，自動照応探索タクティクを用いて自動的に証明探索を行う．自動照応探索タクティクは，destruct_one_ex や destruct_one_pair で DTS の Σ 型およびペア型を分解し[2)]，specialize_H で Π 型の例化を行い，eexists でゴールに対して任意の存在変数を与えて，apply_H でゴールの型に合うものを前提や仮定から適用して証明するものである．

2) destruct_one_ex と destruct_one_pair は，次の Coq 標準ライブラリの中で定義されているものである：https://coq.inria.fr/doc/V8.18.0/stdlib/Coq.Program.Tactics.html

```
Ltac tac1' n  :=
  match n with
  | 0 => cbv in *; intros; try repeat (
    destruct_one_ex || destruct_one_pair ||
    specialize_H || rewrite_H); eexists; try
    apply_H
  | S ?m => move_H' n; cbv in *; intros; try
    repeat (destruct_one_ex || destruct_one_pair
     || specialize_H || rewrite_H); eexists; try
     apply_H
  end.
```

このタクティクを用いることで，テキスト内の全ての可能な先行詞を探索し証明探索結果として登録できる．証明探索結果は Type という型で登録し，(1b) から次のような先行詞を当てはめたテキストが生成される．

**(1b-1)** The lawyer was reluctant to answer the question.

**(1b-2)** The witness was reluctant to answer the question.

## 3 照応探索結果の整合性判定手法

本研究では，上位分類として接続詞ごとに，下位分類として用法ごとに，照応表現の前後の文脈から，到達可能な対象のうち，ある対象が照応表現の指す対象であるか，または照応表現の指す対象でないかを判定する手法を提案する．テストケースとして Winograd Schema Challenge [4] のデータセットを用いている．

### 3.1 because: 原因・理由

**(2)** The trophy doesn't fit into the brown suitcase because it is too small.[3)] ⟹ The suitcase is too small.

**(2-0)** There is a trophy and a brown suitcase.

**(2a)** The trophy doesn't fit into the brown suitcase.

**(2b)** It is too small.

(2-0) は (2) の前件や (2a) の前提である．(2a) because (2b). の談話は，(2b) が原因・理由になり (2a) が成立している．つまり，(2b) ⟹ (2a) が成立するため，整合性判定では (2c) のような推論を行う．

**(2c)** It is too small ⟹ The trophy doesn't fit into the suitcase.

まず，§ 2 での (2-0) を前提とした (2b) の照応探索手続きにより，照応表現 *it* の可能な先行詞を全て探索し，対象を (2b) に当てはめたテキストを生成する．

**(2b-1)** The trophy is too small.

3) [4] の例文 4. より抜粋．

**(2b-2)** The brown suitcase is too small.

次に，生成されたテキストに対し推論を行い，照応探索手続きで生成されたテキストの整合性を判定する．

**(2c-1)** The trophy is too small $\Longrightarrow$ The trophy doesn't fit into the suitcase..

**(2c-2)** The brown suitcase is too small $\Longrightarrow$ The trophy doesn't fit into the suitcase..

推論を行う前に，「スーツケースよりも大きいオブジェクト $x$ は，そのスーツケースの中には入らない．」という常識的な背景知識を仮定に追加する．

```
Hypothesis AnythingBigger :
  forall u: {x: entity & {y:entity &
  {_: suitcase y & is_bigger_than y x}}},
    {y:entity & {_: suitcase y &
    (fit_in y (projT1 u) -> False)}}.
```

最後に，(2c-1), (2c-2) を Theorem として証明を始め，自動証明タクティクを用いて自動的に証明探索を行う．

```
Ltac tac1 := cbv in *; intros; try repeat (
    destruct_one_ex || destruct_one_pair ||
    specialize_H || rewrite_H); try exists_H;
    try apply_H.

Ltac tac4 :=
  solve [repeat (intros; cbv in *; try repeat (
    destruct_one_ex || destruct_one_pair ||
    specialize_H || rewrite_H || destruct_H);
         rewrite_G || exists_H || apply_H ||
    reflexivity || destruct_hyp || simple eapply
     exchange_lemma || info_eauto)].

Ltac tac5 := tac4 || tac1.
```

自動証明タクティクを用いることで，推論を自動的に証明することができる．自動証明タクティクで証明できた場合，推論が正しく，照応探索結果が照応表現の指す対象を同定していることが判定される．証明できなかった場合，推論が正しくない，つまり照応探索結果が照応表現の指す対象を同定していないことが判定される．

## 3.2 but, although: 逆接

**(3)** Sid explained his theory to Mark but he couldn't understand him.[4] $\Longrightarrow$ Mark couldn't understand him.

**(3-0)** There are two mans called Sid and Mark, respectively, and a theory developed by Sid.

**(3a)** Sid explained his theory to Mark.

**(3b)** He couldn't understand him.

4) [4] の例文 48. より抜粋．

"A but B."という談話の特に逆接用法においては，予期の否定 (denial of expectation) [5, 6] および前提された推論 [7] があることが指摘されている．つまり，(3a) but (3b). の談話において，(3b) から (3a) の出来事から予想されることの矛盾が導かれ，(3a) $\wedge$ (3b) $\Longrightarrow \bot$ が成立する．そのため，整合性判定では (3c) のような推論を行う．

**(3c)** Sid explained his theory to Mark and he couldn't understand him. $\Longrightarrow \bot$

まず，§ 2 での照応探索手続きにより，照応表現 *he* の可能な先行詞を全て探索し，対象を (3a) と (3b) に当てはめたテキストを生成する．(3b-1) の照応探索結果は `ThreeB1` として，(3b-2) の照応探索結果は `ThreeB2` として登録した．

**(3b-1)** Sid explained his theory to Mark and Mark couldn't understand him.

**(3b-2)** Sid explained his theory to Mark and Sid couldn't understand him.

推論を行う前に，「$x$ が $y$ に説明するなら，$y$ は $x$ を理解しないということはない」という予期される背景を仮定に追加する．

```
Hypothesis UnderstandExplain :
forall x y : entity, explain y x ->
  ((understand x y -> False) -> False).
```

最後に，(3b-1) $\Longrightarrow \bot$ と (3b-2) $\Longrightarrow \bot$ を Theorem として証明を始め，自動証明タクティクで証明を行う．

```
Theorem Success : ThreeB1 -> False.
Proof.
  tac5.
Qed.

Theorem Failure : ThreeB2 -> False.
Proof.
  tac5.
Abort.
```

(3b-1) から矛盾が導かれ，前後の文脈との整合性が取れていることから，(3b-1) は照応表現の指す対象を同定していることが判定される．一方で，(3b-2) から矛盾が導かれず，前後の文脈との整合性が取れていないことから，(3b-2) は照応表現の指す対象を同定していないことが判定される．

## 3.3 when, after, before: 時制

接続詞 *when* および *after* の意味表示を与えるために，以下では [8, 9] での DTS による時制の分析を簡略化して用いる．まず `time` 型を有理数型の直積 $\mathbb{Q} \times \mathbb{Q}$ として定義し，`time` 型の上にいくつかの関

数および関係を定義する．ここで time 型の要素つまり有理数のペア $(q_1, q_2)$ : time は，ある事態（出来事もしくは状態）の開始時刻 $q_1$ とその継続時間の長さ $q_2$ を表しており，time 型の要素を述語の引数に加えることで，その述語が表す事態の時間を表現する．時間 $t$ : time に対してその終了時刻を表す関数 endtime $t$，および，2 つの時間 $t_1, t_2$ : time に対して「$t_1$ は $t_2$ に先立つ」という事態を表す関係 precede $t_1$ $t_2$ は次のように定義できる．定義の際には有理数型 $\mathbb{Q}$ についての Coq の標準ライブラリ QArith を用いる[5]．

```
Require Import QArith.
Definition time : Type := Q * Q.
Definition endtime : time -> Q :=
    fun t => fst t + snd t.
Definition precede (t1 t2 : time) : Type :=
  endtime t1 < fst t2.
```

次に，[10] での接続詞 *when*, *after* の意味説明を参考にしつつ，これらに対して DTS における意味表示を与える．“When A, B.” という談話において A と B は同じ時刻に終了すると考えられるため，A と B の終了時刻が等しいという条件を意味表示に与える．“A, after B.” という談話においては，B の終了時刻は A の開始時刻に先立つので，precede を用いて B の終了時刻は A の開始時刻に先立つという条件を意味表示に与える．このような意味表示を与えるために，以下の定義では A と B に時間を引数として与える．

```
Definition when (A B : time -> Type) : Type :=
  {t1 : time & {t2 : time & {s : A t1 & B t2 &&
    (endtime t1 = endtime t2)}}}.
Definition after (A B : time -> Type) : Type :=
  {t1 : time & {t2 : time & {s : A t1 & B t2 &&
    (precede t2 t1)}}}.
```

**(4)** Joe paid the detective after he received the final report on the case.[6] $\Longrightarrow$ Joe received the final report.

**(4-0)** There is a detective and a man called Joe.

**(4a)** Joe paid the detective.

**(4b)** He received the final report on the case.

まず，§ 2 での (4-0) を前提とした (4b) の照応探索手続きにより，照応表現 *he* の可能な先行詞を全て探索し，対象を (4b) に当てはめたテキストを生成する．(4b) の照応探索結果を HeReceived として登録した．推論を行う前に，次に，生成されたテキストに対し推論を行い，照応探索手続きで生成されたテキストの整合性を判定する．(4a) after (4b). の談話は，(4b) の終了時刻は (4a) の開始時刻に先立つため，時間関係を取り出しやすくするために補助関数 TimeOfHeReceived を追加し，(4b) $\Longrightarrow$ (4a) に時間関係を追加した推論を行う．

```
Theorem TimeOfHeReceived : HeReceived -> time.
Proof.
  intro.
  destruct X as [x1 [x2 [x3 [x4 x5]]]].
  exact x1.
Defined.

Theorem inference : forall (a: HeReceived),
  {t : time & {y : entity &
  {_ : detective y & {_ : pay t y joe &
  precede (TimeOfHeReceived a) t}}}}.
Proof.
  tac5.
Qed.
```

## 4 おわりに

本研究では，到達可能性をもつ表現が照応表現の先行詞であるか否かを前後の文脈との整合性があるか否かで判定するという手法で実装を行なった．照応探索結果の整合性判定手法では文脈の中でも特に接続詞に着目して，上位分類として接続詞ごとに，下位分類では用法ごとに分類し，because の原因・理由の用法や but, although の逆接の用法および when, after の時制の用法を定式化することで，より多くの照応解析の問題を扱うことができるようになった．

今後の課題としては，照応解析において全ての先行詞の可能性を枚挙するという手続きを Coq タクティクによって可能な限り自動化する点が挙げられる．現段階では，上述の (1b-1) や (1b-2) のような先行詞の可能性に対応するテキストを生成することは部分的に手動で行っている．具体的には，照応表現の指す対象の探索は自動で行えるものの，見つかった対象を照応解析結果として登録してテキストを生成することは，一つ一つの可能性ごとに手動で行っている．Coq がもついくつかのタクティク言語の機能を調べ，この作業もタクティクにより自動化することを目指す[7]．

5） https://coq.inria.fr/doc/V8.18.0/stdlib/Coq.QArith.QArith_base.html

6） [4] の例文 213. より抜粋．

7） 標準的なタクティク言語 Ltac および Ltac2 に加えて例えば Coq-Elpi [11] といったタクティク言語が提案されている．

## 謝辞

本研究は JST CREST JPMJCR20D2 の助成を受けたものである．

## 参考文献

[1] Daisuke Bekki. Proof-theoretic analysis of weak crossover. In **Proceedings of the Eighteenth International Workshop of Logic and Engineering of Natural Language Semantics 18 (LENLS18)**, pp. 75–88, 2021.

[2] Hina Kosaihira, Yuta Takahashi, and Daisuke Bekki. Implementation of Anaphora Resolution Using the Refine Tactic of Coq. Abstract Booklet of the 2023 Workshop on Proof Theory and its Applications, University of Barcelona, July 2023. https://www.ub.edu/prooftheory/event/tps2023workshop/Booklet_of_abstracts_TPSW23.pdf.

[3] The Coq Development Team. The Coq proof assistant reference manual: Version 8.18, 2023. http://coq.inria.fr.

[4] Ernest Davis, Leora Morgenstern, and Charles Ortiz. Winograd Schema Challenge: Collection of Winograd Schemas. https://cs.nyu.edu/~davise/papers/WinogradSchemas/WSCollection.xml, accessed January 10, 2024.

[5] Mitsuko Narita Izutsu. Contrast, concessive, and corrective: Toward a comprehensive study of opposition relations. **Journal of Pragmatics**, Vol. 40, No. 4, pp. 646–675, 2008.

[6] Grégoire Winterstein. What but-sentences argue for: An argumentative analysis of but. **Lingua**, Vol. 122, No. 15, pp. 1864–1885, 2012.

[7] 窪田愛, 佐藤拓真, 天本貴之, 秋吉亮太, 峯島宏次. 逆接の推論関係に着目した日本語談話関係アノテーション. 言語処理学会 第 29 回年次大会 発表論文集, pp. 375–380, 2023.

[8] 宇津木舞香, 戸次大介. 依存型意味論による日本語のテンス・アスペクトの分析に向けて. 2015 年度人工知能学会全国大会（第 29 回）論文集, 2015.

[9] 松岡大樹, 戸次大介, 谷中瞳. 依存型意味論を用いた日本語連体節のテンス解釈. 2023 年度人工知能学会全国大会（第 37 回）論文集, 2023.

[10] Coq semantics for the GF FraCas suite, 2021. https://github.com/GU-CLASP/FraCoq/tree/iwcs2021.

[11] Enrico Tassi. Elpi: an extension language for Coq (Metaprogramming Coq in the Elpi $\lambda$Prolog dialect), 2018. https://inria.hal.science/hal-01637063.