依存型意味論による singular which 疑問文の節埋め込みの分析

船蔵颯
京都大学 人間・環境学研究科
funakura.hayate.28p@st.kyoto-u.ac.jp

概要

本稿では singular which 疑問文の節埋め込みに対して依存型意味論 (Dependent Type Semantics)[1, 2] による分析を提案する。singular which 疑問文は Uniqueness Presupposition と呼ばれる前提を引き起こすことが知られている [3, 4]。また、応答的述語 (responsive predicate) の一つである *know* の補文として singular which 疑問文が埋め込まれるときにも Uniqueness Presupposition は引き起こされる [5]。本稿の分析により、これらの現象が依存型意味論で捉えられることを示す。

1 はじめに

平叙文と疑問文のどちらも補文として埋め込むことができる述語は応答的述語 (responsive predicate) と呼ばれる [6]。例えば *know* は応答的述語のひとつである。

(1) a. John knows Sue danced.
b. John knows which student danced.

さらに、応答的述語は平叙文を補文とする場合と、疑問文を補文とする場合とでそれぞれ異なる意味を持つわけではないことが (2) により示唆される [7]。

(2) Alice knows/realized/reported that Ann left and Bill ~~knows/realized/reported~~ which other girls left.
([7] より引用)

以上を背景として、応答的述語が平叙文と疑問文のいずれも補文としてとりうることを、複数の意味を仮定せずに捉えることは、形式意味論における課題の一つとなっている [7]。

上述の課題に関連する現象として、疑問文埋め込み構文 *x Vs Q.* において引き起こされる前提の問題が挙げられる。singular which 疑問文 *which N* φ *?*は *Exactly one N* φ*.* を前提とする [3, 4]。また、*know* が singular which 疑問文を補文として埋め込むときにも同様の前提が観察される [5]。singular which 疑問文が引き起こすこのような前提は Uniqueness Presupposition と呼ばれ、疑問文埋め込みの意味論で取り組まれている現象の一つである [4, 5][1)]。Uniqueness Presupposition の例を (3) に示す。

(3) a. Which student danced?
presupposes Exactly one student danced.
b. John knows which student danced.
presupposes Exactly one student danced.

ここでいう singular which 疑問文は、疑問詞 *which* の直後に単数形の名詞が現れる形の疑問文である。

依存型意味論は Martin-Löf の依存型理論 [8] に基づく自然言語の証明論的意味論である。依存型意味論による先行の分析として、疑問文については [9]、応答的述語 *know* を含む叙実述語 (factive predicate) への平叙文埋め込みについては [10] で分析が行われているが、著者の知る限りでは singular which 疑問文とその埋め込みの分析は未着手の問題である。

本稿では singular which 疑問文に対して依存型意味論 (Dependent Type Semantics)[1, 2] による意味表示を提案する。また、提案した意味表示のもとで (3) の Uniqueness Presupposition が予測されることを示す (5.1 節)。さらに、平叙文を補文とする *know* の意味表示 [10] により、singular which 疑問文を補文とする場合の Uniqueness Presupposition が予測されることを示す (5.2 節)。

2 現象の記述

本章では、本稿で主な説明対象とする現象について述べる。そのうちの一つは singular which 疑問文の引き起こす Uniqueness Presupposition であり、もう一つは singular which 疑問文が *know* の補文となると

1) この前提を語用論的前提とみなす見方もある [5]。本稿では Uniqueness Presupposition を意味論的な前提とみなし、したがって意味表示に含めることとする。

きに尚も観察される Uniqueness Presupposition である。

2.1 singular which 疑問文の前提

singular which 疑問文 *which N φ ?*は *Exactly one N φ.* を前提とする [3, 4]。

?? Which student danced?
presupposes Exactly one student danced.

1 章でも述べた通り、この前提は Uniqueness Presupposition と呼ばれ、以下では UP と略記する。

2.2 叙実述語への疑問文埋め込みによる前提

上に述べた UP は叙実述語への疑問文埋め込み *x knows which N φ.* においても観察される [5]。次に示すように、*x knows which N φ.* は *Exactly one student danced.* を含意する。

(4) John knows which student danced.
entails Exactly one student danced.

さらに (4) の後件 *Exactly one student danced.* は、否定、疑問、条件文の前件といった構文のもとでも含意される。すなわち次が成り立つ。

(5) a. John doesn't know which student danced.
b. Does John know which student danced?
c. If John knows which student danced, that's a problem.
entails Exactly one student danced.

このように否定、疑問、条件文の前件といった構文のもとでも含意が保たれる現象を投射 (projection) といい、投射が起こるような含意を特別に前提という [11]。以上から *Exactly one N φ.* は *x knows which N φ.* の前提である。

平叙文の叙実述語への埋め込み構文においても同様の前提が観察される [12]。その例を次に示す。

(6) John knows that Sue danced.
presupposes Sue danced.

以上から、singular which 疑問詞の意味論には少なくとも以下のことが求められる。

1. singular which 疑問文の UP を予測する
2. *know* が平叙文を補文とする構文 *x knows [declarative].* と、疑問文を補文とする構文 *x knows [interrogative].* それぞれの前提を、*know* に複数の意味を仮定せずして予測する

3 依存型意味論

本章では依存型意味論の理論的基礎である依存型理論の基本的なアイデアを述べ、その後に依存型意味論に固有の特徴の一つである未指定項による前提の取り扱いについて述べる。

3.1 依存型理論

依存型理論は単純型理論に対して Π 型、Σ 型などを加えたものであり、大きな特徴の一つとして、型の中に項が現れることができる。また、それぞれの型の意味は推論規則によって与えられる。本稿で用いる型の推論規則は付録に掲載している。

カリー・ハワード対応により、型は命題、項は命題の証明と同一視できる。型と命題の対応関係の一部を表 1 に示す。ただし、Σ 型の表記として依存型意味論に固有の表記 $\begin{bmatrix} x : A \\ B(x) \end{bmatrix}$ を用いている[2]。

表 1 型と命題の対応

依存型意味論における型	対応する命題
$(\Pi x : A)B(x)$	$(\forall x \in A)B(x)$
$A \to B$	$A \supset B$
$\begin{bmatrix} x : A \\ B(x) \end{bmatrix}$	$(\exists x \in A)B(x)$
$A \times B$	$A \& B$

依存型意味論では、自然言語の意味表示に依存型理論の型を用いる。また、統語論として典型的には組み合わせ範疇文法 (Combinatory Categorial Grammar)[13] が採用され、これは本稿でも同様である。

3.2 未指定項

依存型意味論は上述のような性質を備える依存型理論に対して、前提・照応を表現する演算子として未指定項 $@_i$ を導入する。i は自然数であり、複数の未指定項を区別するための添字である。依存型意味論においては、前提は文の意味表示に対する型検査を通して予測される。

2) 依存型意味論では Π 型の表記として $(x : A) \to B(x)$ がしばしば用いられるが、本稿ではスペースの都合により [8] の表記 $(\Pi x : A)B(x)$ を用いる。

$$\dfrac{\dfrac{\text{John}}{\begin{array}{c}NP:\\ john\end{array}} \quad \dfrac{\dfrac{\text{knows}}{\begin{array}{c}(S\backslash NP)/\bar{S}:\\ \lambda p.\lambda x.\lambda c.\mathbf{kn}(x)(pc)(@_i c)\end{array}} \quad \dfrac{\dfrac{\text{that}}{\begin{array}{c}\bar{S}/S:\\ \lambda P.P\end{array}}\quad \dfrac{\text{Sue danced}}{\begin{array}{c}S:\\ \lambda c.\mathbf{d}(sue)\end{array}}}{\begin{array}{c}\bar{S}\\ \lambda c.\mathbf{d}(sue)\end{array}}>}{\begin{array}{c}S\backslash NP\\ \lambda x.\lambda c.\mathbf{kn}(x)(\mathbf{d}(sue))(@_i c)\end{array}}>}{\begin{array}{c}S:\\ \lambda c.\mathbf{kn}(john)(\mathbf{d}(sue))(@_i c)\end{array}}<$$

図 1　*John knows Sue danced.* の意味表示導出

$$(\Pi I)\dfrac{(\Pi E)\dfrac{(\Pi E)\dfrac{(\Pi E)\dfrac{(CON)\dfrac{}{\mathbf{kn}:\mathsf{Entity}\to(\Pi \mathsf{P}:\mathsf{type})(\mathsf{P}\to\mathsf{type})}\quad (CON)\dfrac{}{john:\mathsf{Entity}}}{\mathbf{kn}(john):(\Pi \mathsf{P}:\mathsf{type})(\mathsf{P}\to\mathsf{type})}\quad (\Pi E)\dfrac{(CON)\dfrac{}{sue:\mathsf{Entity}}\quad (CON)\dfrac{}{\mathbf{d}:\mathsf{Entity}\to\mathsf{type}}}{\mathbf{d}(sue):\mathsf{type}}}{\mathbf{kn}(john)(\mathbf{d}(sue)):\mathbf{d}(sue)\to\mathsf{type}}\quad (\Pi E)\dfrac{@_i:\delta\to\mathbf{d}(sue)\quad \dfrac{}{c:\delta}(1)}{@_i c:\mathbf{d}(sue)}}{\mathbf{kn}(john)(\mathbf{d}(sue))(@_i c):\mathsf{type}}}{\lambda c.\mathbf{kn}(john)(\mathbf{d}(sue))(@_i c):\delta\to\mathsf{type}}(1)$$

図 2　意味表示 (8) の型検査

例として、叙実述語が平叙文を補文とするときの前提 (7) の依存型意味論による予測 [10] について述べる。

(7) John knows Sue danced.
presupposes Sue danced.

組み合わせ範疇文法 (CCG) による図 1 の導出により、(7) の前件 *John knows Sue danced.* には次の意味表示が与えられる。

(8) $\lambda c.\mathbf{kn}(john)(\mathbf{d}(sue))(@_i c)$

依存型意味論では、文、すなわち CCG カテゴリが S である表現の意味表示は $\delta\to\mathsf{type}$ という型を持つことが要求される。この条件は文の felicity condition と呼ばれる [1]。意味表示 (8) が felicity condition を満たすことの証明は図 2 の通りである。この証明では次の (9) が閉じていない葉として残っている。

(9) $@_i:\delta\to\mathbf{d}(sue)$

付録の規則 ($@F$) より、(9) を証明するためには先行する談話の証明から $\mathbf{d}(sue)$ の証明、つまり、*Sue danced.* の証明を構成しなければならない。以上の手続きにより、前提推論 (7) が成り立つことが予測される。

4 依存型意味論における singular which 疑問詞の分析

4.1 叙実述語への疑問文埋め込みによる前提

singular which 疑問詞に対して、次の意味表示を提案する。

(10) $\lambda p.\lambda q.\lambda c.q\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (p(x)(c)\times q(x)(c)\times\\ (\Pi y:\mathbf{Entity})(p(y)(c)\times q(y)(c)\\ \to(y=_{\mathbf{Entity}}x))) \end{bmatrix}\right)\right)$

(10) に現れる $@_i c :: A$ という形の項は型アノテーション (type annotation) と呼ばれ、項 $@_i c$ の持つ型が A であるとあらかじめ指定する際に用いられる。

図 3 の導出木により、singular which 疑問文 (11-a) の意味表示として (11-b) が得られる。

(11) a. Which student danced?

b. $\lambda c.\mathbf{d}\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\mathbf{s}(x)\times\mathbf{d}(x)\times\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\\ \to(y=_{\mathbf{Entity}}x))) \end{bmatrix}\right)\right)$

この表示により、2 章で提示した事実が説明されることを次章にて示す。

5 分析の経験的な検証

5.1 singular which 疑問文の前提

本節では、singular which 疑問文の UP が本稿の分析により予測されることを示す。

singular which 疑問文の意味表示 (11-b) が felicity condition を満たすことは図 4 により示される。この証明では次の (12) が閉じていない葉として残っている。

(12) $@_i:\delta\to\begin{bmatrix} x:\mathbf{Entity}\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\to x=_{\mathbf{Entity}}y) \end{bmatrix}$

3.2 節での議論と同様に、(11-b) が felicity condition を満たすためには、先行する談話の証明から *Exactly one student danced.* の証明を構成しなければならない。以上のようにして、singular which 疑問文の UP

$$
\dfrac{
  \dfrac{
    \dfrac{
      \dfrac{\text{Which}}{(S/NP)/NP: \lambda p.\lambda q.\lambda c.q\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (p(x)(c)\times q(x)(c)\times\\ (\Pi y:\mathbf{Entity})(p(y)(c)\times q(y)(c)\to (y=_{\mathbf{Entity}} x)))\end{bmatrix}\right)\right)}
      \quad
      \dfrac{\text{student}}{NP: \lambda x.\lambda c.\mathbf{s}(x)}
    }{S/NP: \lambda q.\lambda c.q\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\mathbf{s}(x)\times q(x)(c)\times\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times q(y)(c)\to (y=_{\mathbf{Entity}} x)))\end{bmatrix}\right)\right)}{>}
    \quad
    \dfrac{\text{danced}}{NP: \lambda x.\lambda c.\mathbf{d}(x)}
  }{S: \lambda c.\mathbf{d}\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\mathbf{s}(x)\times \mathbf{d}(x)\times\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times \mathbf{d}(y)\to (y=_{\mathbf{Entity}} x)))\end{bmatrix}\right)\right)}{>}
  \quad
  \dfrac{?}{S\backslash S: \lambda x.x}
}{S: \lambda c.\mathbf{d}\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\mathbf{s}(x)\times \mathbf{d}(x)\times\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times \mathbf{d}(y)\to (y=_{\mathbf{Entity}} x)))\end{bmatrix}\right)\right)}{<}
$$

図 3　*Which student danced?*の意味表示導出

$$
{\scriptstyle (\Pi I)}\dfrac{
  {\scriptstyle (\Pi E)}\dfrac{
    {\scriptstyle (CON)}\dfrac{}{\mathbf{d}:\mathbf{Entity}\to\mathsf{type}}
    \quad
    {\scriptstyle (\Sigma E)}\dfrac{
      {\scriptstyle (\Pi E)}\dfrac{
        @_i:\delta\to\begin{bmatrix} x:\mathbf{Entity}\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\to x=_{\mathbf{Entity}} y)\end{bmatrix}
        \quad
        \dfrac{}{c:\delta}{\scriptstyle (1)}
      }{@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\to x=_{\mathbf{Entity}} y)\end{bmatrix} : \begin{bmatrix} x:\mathbf{Entity}\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\to x=_{\mathbf{Entity}} y)\end{bmatrix}}
    }{\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\to x=_{\mathbf{Entity}} y)\end{bmatrix}\right) : \mathbf{Entity}}
  }{\mathbf{d}\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\to x=_{\mathbf{Entity}} y)\end{bmatrix}\right)\right) : \mathsf{type}}
}{\lambda c.\mathbf{d}\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\to x=_{\mathbf{Entity}} y)\end{bmatrix}\right)\right) : \delta\to\mathsf{type}}{\scriptstyle (1)}
$$

図 4　意味表示 (11-b) の型検査

が予測される。

5.2 叙実述語への疑問文埋め込みによる前提

本節では次の前提推論が、本稿の提案により予測されることを示す。

(13) John knows which student danced.
presupposes Exactly one student danced.

(13) の前件 *John knows which student danced.* の意味表示として次の (14) が得られる。

(14) $\mathbf{kn}(john)\left(\mathbf{d}\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\mathbf{s}(x)\times\mathbf{d}(x)\times\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\\ \to (y=_{\mathbf{Entity}} x)))\end{bmatrix}\right)\right)\right)(@_i c)$

平叙文埋め込みの場合 (図 2) と同様に、**kn** の第二引数は型 type を持たなければならない。したがって次が成り立つ必要がある。

(15) $\mathbf{d}\left(\pi_1\left(@_i c :: \begin{bmatrix} x:\mathbf{Entity}\\ (\mathbf{s}(x)\times\mathbf{d}(x)\times\\ (\Pi y:\mathbf{Entity})(\mathbf{s}(y)\times\mathbf{d}(y)\\ \to (y=_{\mathbf{Entity}} x)))\end{bmatrix}\right)\right) : \mathsf{type}$

(15) を示す証明は図 4 の部分証明図であることから、この場合にも (12) が閉じていない前提として残る。したがって、(14) が felicity condition を満たすには、先行する談話の証明から *Exactly one student danced.* の証明を構成しなければならない。以上のようにして、*know* への疑問文埋め込みによる UP が予測される。

6 おわりに

本稿では singular which 疑問文に対して依存型意味論による意味表示を提案した。本稿で与えた意味表示の下では、singular which 疑問文の UP を自然に予測することができる。また、*know* が singular which 疑問文を補文とする場合の UP を、平叙文を補文とする *know* と共通の意味表示の下で予測することができる。これは、平叙文と補文とする *know* と、疑問文を補文とする *know* が共通の意味を持つという仮説に沿うものである。

singular which 疑問文以外の which 疑問文へと分析を拡張することは今後の課題の一つである。本稿では応答的述語の一つである *know* に関係する前提現象を議論の対象としている。応答的述語の意味論へ向けたその他の課題として、例えば Predicates of relevance と呼ばれる述語 (*care*, *matter* など) の前提パターン [14] がある。これらの述語には、補文が平叙文であるときに引き起こされる前提が、補文が疑問文であるときには観察されないという特徴がある。こうした述語を依存型意味論により捉えることも今後取り組むべき課題である。

謝辞

本稿執筆を通して複数回にわたって議論していただいた京都大学人間・環境学研究科の櫻川貴司准教授に感謝いたします。

参考文献

[1]Bekki Daisuke. Representing anaphora with dependent types. In *International conference on logical aspects of computational linguistics*, pp. 14–29. Springer, 2014.

[2]Bekki Daisuke and Mineshima Koji. Context-passing and underspecification in dependent type semantics. In *Modern perspectives in type-theoretical semantics*, pp. 11–41. Springer, 2017.

[3]Duží Marie and Číhalová Martina. Questions, answers, and presuppositions. *Computación y Sistemas*, Vol. 19, No. 4, pp. 647–659, 2015.

[4]Dayal Veneeta. *Locality in Wh-quantification: Questions and relative clauses in Hindi*. Kluwer Academic Publishers, 1996.

[5]Uegaki Wataru. The existential/uniqueness presupposition of wh-complements projects from the answers. *Linguistics and Philosophy*, Vol. 44, No. 4, pp. 911–951, 2021.

[6]Lahiri Utpal. *Questions and Answers in Embedded Contexts*. Oxford University Press UK, 2001.

[7]Uegaki Wataru. The semantics of question-embedding predicates. *Language and Linguistics Compass*, Vol. 13, No. 1, p. e12308, 2019.

[8]Martin-Löf Per. *Intuitionistic Type Theory*. Bibliopolis, 1984.

[9]Watanabe Kazuki, Mineshima Koji, and Bekki Daisuke. Questions in dependent type semantics. In *Proceedings of the Sixth Workshop on Natural Language and Computer Science*, pp. 23–33, Gothenburg, Sweden, May 2019. Association for Computational Linguistics.

[10]Tanaka Ribeka, Mineshima Koji, and Bekki Daisuke. Factivity and presupposition in dependent type semantics. *Journal of Language Modelling*, Vol. 5, No. 2, p. 385–420, 2017.

[11]*The Cambridge Handbook of Formal Semantics*. Cambridge Handbooks in Language and Linguistics. Cambridge University Press, 2016.

[12]Wilson Deirdre. Presuppositions on factives. *Linguistic Inquiry*, Vol. 3, No. 3, p. 405–10, 1972.

[13]Steedman Mark. *The syntactic process*, Vol. 24. MIT press Cambridge, MA, 2000.

[14]Elliott Patrick D., Klinedinst Nathan, Sudo Yasutada, and Uegaki Wataru. Predicates of Relevance and Theories of Question Embedding. *Journal of Semantics*, Vol. 34, No. 3, pp. 547–554, 04 2017.

付録

本稿で用いた型および未指定項に関する推論規則の一部を掲載する。ただし i, j, k は自然数であるとする。詳細は [1, 2] を参照されたい。

$$(@F)\ \dfrac{A : \mathtt{type}_i \qquad A \quad \mathit{true}}{@_j : A} \qquad (ann)\ \dfrac{t : A}{(t :: A) : A}$$

$$(\Pi F)\ \dfrac{A : \mathtt{type}_i \qquad \begin{array}{c}\overline{x : A}^{\,(k)} \\ \vdots \\ B(x) : \mathtt{type}_j\end{array}}{(\Pi x : A)B(x) : \mathtt{type}_{\max(i,j)}}\,(k) \qquad (\Pi I)\ \dfrac{A : \mathtt{type}_i \qquad \begin{array}{c}\overline{x : A}^{\,(k)} \\ \vdots \\ B(x) : M\end{array}}{\lambda x.M : (\Pi x : A)B(x)}\,(k) \qquad (\Pi E)\ \dfrac{M : (\Pi x : A)B(x) \qquad N : A}{MN : B(x)[N/x]}$$

$$(\Sigma F)\ \dfrac{A : \mathtt{type}_i \qquad \begin{array}{c}\overline{x : A}^{\,(k)} \\ \vdots \\ B(x) : \mathtt{type}_j\end{array}}{\begin{bmatrix} x : A \\ B(x) \end{bmatrix} : \mathtt{type}_{\max(i,j)}}\,(k) \qquad (\Sigma I)\ \dfrac{M : A \qquad N : B(x)[M/x]}{(M, N) : \begin{bmatrix} x : A \\ B(x) \end{bmatrix}}$$

$$(\Sigma E)\ \dfrac{M : \begin{bmatrix} x : A \\ B(x) \end{bmatrix}}{\pi_1(M) : A} \qquad (\Sigma E)\ \dfrac{M : \begin{bmatrix} x : A \\ B(x) \end{bmatrix}}{\pi_2(M) : B(x)[\pi_1(M)/x]}$$

$$(=F)\ \dfrac{A : \mathtt{type}_i \qquad M : A \qquad N : A}{M =_A N : \mathtt{type}_i} \qquad (=I)\ \dfrac{A : \mathtt{type}_i \qquad M : A}{\mathsf{r}_A(M) : M =_A M} \qquad (=E)\ \dfrac{p : M =_A M'}{M = M' : A}$$