3H2-OS3-6

語彙概念構造を用いた語彙間意味関係グラフの自動構築

Automatic Construction of Lexical Semantic Relation Graph using Lexical Conceptual Structures

松林 優一郎*1　宮尾 祐介*1　相澤 彰子*1
Yuichiroh Matsubayashi　Yusuke Miyao　Akiko Aizawa

*1国立情報学研究所 コンテンツ科学研究系
Digital Content and Media Sciences Research Division, National Institute of Informatics

Linguistic resources that describe semantic relations between lexicons in the level of argument structures, such as FrameNet, are considered as useful resources for technologies of semantic analysis including paraphrasing and recognizing textual entailments. However, semantic relations in existing resources were annotated by human effort. Moreover, maintaining consistency of whole relation graph costs much. We propose a formal theory for constructing a semantic relation graph of lexicons by extending a theory of lexical conceptual structure.

1. はじめに

近年の語彙の言い換えや含意関係理解などの語彙間の意味関係を利用する研究では、多くの場合、シソーラスが利用されてきたが、それらの資源の作成方法は、人手による関係記述か、もしくは国語辞典等に基づく自動抽出の方法が殆どであった [鍛治 03]。語彙の言い換えにおいては、シソーラスのような語彙そのものの直接的な意味的関係のみならず、対象とする語彙の項同士の意味的関係についても記述することが必要であるが、そのような細やかな関係記述を、人手で、あるいは非形式的な意味表現から定義するのは困難である。

本研究では、語彙に対して予め与えておいた形式的な意味記述から、語彙間の意味関係を導出する理論的な枠組みを提案する。具体的には、我々の手法は、語彙概念構造（LCS）の理論に基いて、語彙の意味を少数の意味的述語の入れ子構造として記述し、二つの構造の間の意味的関係を定義する関係規則を導入することで、異なる LCS 間の意味的関係を形式的に定義する。

2. 語彙概念構造（LCS）

本稿が提案する枠組みでは、語彙の意味の記述形式として、語彙概念構造（LCS）を利用する。語彙概念構造 [Jackendoff 90] は、語彙が表現する出来事や状態を、汎化された意味の構造として記述するための体系である。語彙の意味は、複数の基本述語によって分解され、その組み合わせ構造で表現される。例えば、図 1 は我々の LCS 辞書中の動詞「投げる」の、ある語義についての概念構造を表しているが、この中には cause, affect, go, away_from, toward, locate, in, at といった述語が含まれている。LCS 内の各項は、語彙が取る文中の項によって埋められる。図 1 の例文の場合、i, j, k が、それぞれ「太郎」「ボール」「窓」によって埋められる。

LCS の基本述語は、その組み合わせによって、動詞が表現する、行為－変化－状態という連鎖の部分または全部を表現出来るように設計されている。従って、用いられる各基本述語は、この連鎖における形式的な意味的機能を表す。例えば、図 1 の throw の例では、cause, affect, go, away_from, from, toward の複合により、行為（i が j に作用）－変化－状態（j が、$i \to k \to l$ と移動）の全体が表現されている。

$$\left[\begin{array}{l}\text{cause}(\text{affect}(i,j),\ \text{go}(j,\ \left[\begin{array}{l}\text{away\_from}(\text{locate}(\text{in}(i)))\\ \text{from}(\text{locate}(\text{at}(k)))\\ \text{toward}(\text{locate}(\text{at}(l)))\end{array}\right]))\end{array}\right]$$

・[太郎 $_i$] が [ボール $_j$] を [窓 $_k$] から投げた。

図 1: 動詞「投げる」の語彙概念構造とその例文

表 1: 主要な基本述語とその意味機能

述語名	意味機能
state(x, y)	第一項が、第二項で指定される状態にある。
cause(x, y)	第一項で指定される行為が、第二項で指定される変化を引き起こす。
act(x)	第一項が、自身に作用する活動をする。
affect(x, y)	第一項が、第二項に作用を与える。
react(x, y)	第一項が、第二項の影響を受けて、自身に作用する活動をする。
go(x, y)	第一項が、第二項で指定される経路に沿って変化する。
away_from(x)	変化の始点
from(x)	変化の始点の方向
via(x)	変化の経由点
toward(x)	変化の着点の方向
to(x)	変化の着点
along(x)	変化の経路となる連続的な対称

本稿では、特に動詞及び事態性名詞の意味関係について取り扱う。本稿で我々が用いている LCS 体系は、その大部分が Jackendoff の理論 [Jackendoff 90] に基づく。但し、実世界の多様な語彙を整合性を保ちながら記述可能にするために、いくつかの細かな拡張を加えている。我々の拡張した LCS 体系は図 2 に与えるが、より具体的な説明については、[松林 11] を参照されたい。

3. 概念構造上の関係規則

一度語彙の意味が概念構造で記述されると、それらの語彙は構造の差異による比較が可能になる。それぞれの語彙に正しく概念構造が割り当てられているならば、それらの構造は、(LCS で記述することが可能な粒度の範囲で) 同義なら同じ構造に、類義なら類似の構造になるはずである。

そこで、我々は、語彙間の意味的関係を概念構造の書き換え

連絡先: 氏名：松林優一郎，所属：国立情報学研究所，住所：東京都千代田区一ツ橋 2-1-2，電話：03-4212-2667，E-mail：y-matsu@nii.ac.jp

表 2: LCS 構造の書換え規則と意味関係（抜粋）。書き換え前が子、書き換え後が親の LCS 構造

大分類	規則の種類	構造の差	意味的関係性
式単位の操作	式の削除	LCS から comb[EVENT] を一つ削除する	子の事実が成り立つとき、親の事実が成り立つ
	式の入れ替え	EVENT の式と comb[EVENT] の式を一つ交換する	親と子は注目点が異なるのみで同一のイベント
	式の副構造化	LCS に EVENT 式が二つ以上あるとき、EVENT の一つを comb[EVENT] として書き換える	親と子は注目点が異なるのみで同一のイベント
部分構造の取り立て	目的の取り立て	ある purpose の内部の EVENT のみで新しい LCS を作る	子の事実が成り立つとき、親の事実が起こることが想定される。
	変化の取り立て	ある cause の式を内部の go の構造だけに書き換える	子の動作が、親の変化を引き起こす
	状態の取り立て	ある go の式を、path 内部の to の部分を取り出し、state 式として書き換える	子の変化が、親の状態を引き起こす
式内部の変換	定数項の変数化	ある定数項を変数項に書き換える	子の事実が成り立つとき、子と同じ構造になるように変数項を埋めた親の事実が起こる
	位置を方向に変換	PATH 内の away_from や to を from, toward に書き換える	子の事実が成り立つとき、親の事実が起こる
	項の単一化の解除	LCS 中に二回以上出現するある変数項を未出現の新しい変数項に書き換える	子の事実が成り立つとき、子と同じ構造になるように変数項を埋めた親の事実が起こる

$$\mathrm{LCS} = \begin{bmatrix} \mathrm{EVENT}+ \\ \mathrm{comb}\big[\mathrm{EVENT}\big]* \end{bmatrix} \qquad \mathrm{EVENT} = \begin{bmatrix} \mathrm{CORE} \\ \mathrm{MODALS} \end{bmatrix}$$

$$\mathrm{CORE} = \left\{ \begin{array}{l} \mathrm{state(arg,\ STATE)} \\ \mathrm{go(arg,\ PATH)} \\ \mathrm{cause(act(arg1),\ go(arg1,\ PATH))} \\ \mathrm{cause(affect(arg1,\ arg2),\ go(arg2,\ PATH))} \\ \mathrm{cause(react(arg1,\ arg2),\ go(arg1,\ PATH))} \end{array} \right\}$$

$$\mathrm{MODALS} = \begin{bmatrix} \mathrm{manner(constant)?} \\ \mathrm{mean(constant)?} \\ \mathrm{instrument(constant)?} \\ \mathrm{purpose(EVENT \mid arg)*} \end{bmatrix} \qquad \mathrm{PATH} = \begin{bmatrix} \mathrm{away\_from(STATE)?} \\ \mathrm{from(STATE)?} \\ \mathrm{via(STATE)?} \\ \mathrm{toward(STATE)?} \\ \mathrm{to(STATE)?} \\ \mathrm{along(arg)?} \end{bmatrix}$$

$$\mathrm{STATE} = \left\{ \begin{array}{l} \mathrm{be} \\ \mathrm{locate(PLACE)} \\ \mathrm{orient(PLACE)} \\ \mathrm{extent(PLACE)} \\ \mathrm{connect(arg)} \end{array} \right\} \qquad \mathrm{PLACE} = \left\{ \begin{array}{l} \mathrm{in(arg)} \\ \mathrm{on(arg)} \\ \mathrm{cover(arg)} \\ \mathrm{fit(arg)} \\ \mathrm{inscribed(arg)} \\ \mathrm{beside(arg)} \\ \mathrm{around(arg)} \\ \mathrm{near(arg)} \\ \mathrm{inside(arg)} \\ \mathrm{at(arg)} \end{array} \right\}$$

図 2: 拡張された LCS の記述体系*2。 演算子 $+, *, ?$ は一般的な正規表現の文法に従う。 $|, \{\}$ は選択を表す。

によって定義した。この考え方では、まず、LCS 構造上の基本的な意味的関係を決める原始的な書き換え規則を定義し、複雑な LCS 構造同士の意味関係は、それらの書き換え規則の組み合わせによって結び付くものとする。表 2 に、我々の定義する 1 3 種類の原始的な関係規則のうちのいくつかを抜粋した。これらは、大きく、式単位の操作、部分構造の取り立て、部分構造の変換の三つに分けられる。式単位の操作は、式の削除、主構造式と副構造式 (comb の式) の交換などがあり、取り立てには、変化の取り立て、状態の取り立て、目的の取り立てがあり、部分構造の変換には、定数項の変数化や項の単一化の解除、位置から方向への変換などがある。「式の入れ替え」以外の書き換え規則には向きがあり、書き換え前を子、書き換え後を親として、基本的に親のほうが抽象的、あるいは汎用的な語彙を表す。

これらの規則を利用して、語彙と語彙の意味関係を定義付けるため、図 3 のアルゴリズムを利用して、LCS 構造をノード、関係規則の種類をエッジとした、語彙の意味関係グラフを作成する。アルゴリズムは、予め用意しておいた LCS 辞書中の全ての異なる LCS 構造について、再帰的に書き換え規則を適用しながら仮想的な LCS 構造のノードを生成し、グラフに

*2 簡単のため、各述語の細かな属性値は省略してある。

```
グラフ G の初期ノード：辞書中の語彙の異なる LCS 構造
グラフ G の初期エッジ：空集合

G の各 LCS 構造 s について{
  generate_node(s)
}

generate_node(s){
  それぞれの書き換え規則 r について{
    s に r が適用可能ならば{
      s に r を適用し、s' を生成
      G にエッジ (s, s', r) を追加
      もし、s' が G になければ{
        G にノード s' を追加
        generate_node(s')
      }
    }
  }
}
```

図 3: 語彙間意味関係グラフを生成するアルゴリズム

追加する。ある語彙に対応する LCS 構造から再帰的に生成されたノードが、辞書中の語彙の LCS 構造と一致するとき、これらの語彙に意味的な含意関係があるとみなせることになる。

我々の定義した LCS 上の関係規則とアルゴリズムによって生成されるグラフの一部を図 4 に表した。矢印は先祖から子孫への方向を表しており、アルゴリズム自体は、各語彙から始まって、親方向に進むことに注意されたい。また、いくつかの中間にある仮想ノードは省略されている。語彙の中には、「受け取る」と「買う」のように、直接的な先祖－子孫関係で結びつくものもあれば、「入る」と「受け取る」のように、上位の仮想的なノードを通じて間接的に意味の繋がりを持つものも存在する。

4. 関連研究

LCS を用いた言い換えの研究として、藤田らの研究がある [藤田 06]。彼らの方法は、機能動詞構文や態の変換を主に扱っている。一方、我々の理論は、文中で主体となる動詞性語彙の意味関係を扱うという点でこれと異なっている。

5. まとめ

本稿では、語彙概念構造の理論を拡張することで、事前に与えられた動詞（及び事態性名詞）の LCS 構造から語彙間の

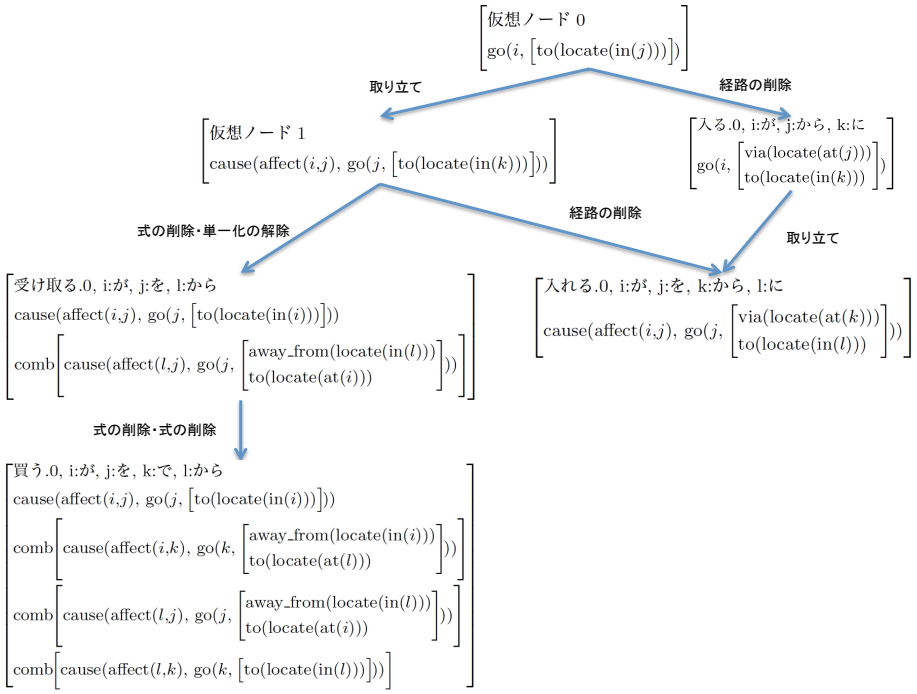

図 4: 関係規則により定義付けられる語彙の意味関係の一例

意味関係を計算する枠組みを提案した。LCS 構造間の関係を、書き換え規則による遷移として定義することで、含意関係を持つ語彙動詞を結びつける関係グラフを生成することを可能にした。但し、現在の我々の LCS の枠組みでは、「買う」と「輸入する」などの、意味の区別出来ない語彙も存在するため、このような語彙に関して、各変数項に選択選好の型を用意するなど、拡張的な理論を今後用意していくことが必要である。

参考文献

[Jackendoff 90] Jackendoff, R.: *Semantic Structures*, The MIT Press (1990)

[松林 11] 松林 優一郎, 宮尾 祐介, 相澤 彰子：語彙概念構造による意味役割の形式化と複数役割の割り当て, 言語処理学会第 17 回年次大会予稿集 (2011)

[藤田 06] 藤田 篤, 降幡 建太郎, 乾 健太郎, 松本 祐治：語彙概念構造に基づく言い換え生成：機能動詞構文の言い換えを例題に, 情報処理学会論文誌, Vol. 47, No. 6, pp. 1963–1975 (2006)

[鍛治 03] 鍛治 伸裕, 河原 大輔, 黒橋 禎夫, 佐藤 理史：格フレームの対応付けに基づく用言の言い換え, 自然言語処理, Vol. 10, No. 4, pp. 65–81 (2003)